Dipartimento di Informatica e
Scienze dell'Informazione

Object Representation and Identification in Image Sequences

Francesca Odone

Theses Series **DISI-TH-2002-05**

DISI, Università di Genova
v. Dodecaneso 35, 16146 Genova, Italy
http://www.disi.unige.it/

Università degli Studi di Genova

Dipartimento di Informatica e Scienze dell'Informazione

Dottorato di Ricerca in Informatica

Ph.D. Thesis in Computer Science

Object Representation and Identification in Image Sequences

Francesca Odone

May, 2002

Dottorato di Ricerca in Informatica
Dipartimento di Informatica e Scienze dell'Informazione
Università degli Studi di Genova

DISI, Univ. di Genova
via Dodecaneso 35
I-16146 Genova, Italy
http://www.disi.unige.it/

Ph.D. Thesis in Computer Science

Submitted by Francesca Odone
DISI, Univ. di Genova
odone@disi.unige.it

Date of submission: December 2001

Title: Object Representation and Identification in Image Sequences

Advisor: Prof. Alessandro Verri
Dipartimento di Informatica e Scienze dell'Informazione
Università di Genova
verri@disi.unige.it

Ext. Reviewers: Prof. Vittorio Murino
Dipartimento Scientifico e Tecnologico
Università di Verona
murino@sci.univr.it

Prof. Fabrizio Smeraldi
School of Information Science, Computer and Electrical Engineering
University of Halmstad,
fabrizio.smeraldi@ide.hh.se

Abstract

This thesis considers the representation and the identification of objects in image sequences: objects are represented by information extracted from image sequences, and this representation is exploited to solve problems such as object detection, recognition, and scene location.
To achieve these goals, a number of issues had to be taken into account, primarily,

- Deal with redundancy in image sequences,
- Estimate the similarity between images,
- Formalize the object representation problem.

This thesis produces original contributions in the following areas:

Image sequence registration and mosaic construction: I present a robust sequence registration technique and apply it to both mosaic construction and mosaic-based multiple motion segmentation. I also show how it can be used to enhance the quality of video sequences, or to perform video compression, thus reducing the redundancy of sequences.

Evaluation of similarity between grey-level images: using images to identify or recognize an object in a scene raises the problem of checking whether a particular object appears in a certain image. If an image-based representation of the object is chosen, one must compare images or parts of them, and evaluate their degree of similarity. For this purpose, a multiscale similarity method based on the definition of the Hausdorff distance has been devised. It is tolerant to scene variations (which may induce occlusions), to small light variations, and performs well even in the presence of small geometric deformations.

Kernel methods for object representation: identification and recognition problem are often cast in the more general framework of binary classification. Kernel methods offer an alternative approach when linear classification is inadequate, by performing a mapping of data points to a high dimensional space and exploiting *a priori* knowledge to choose more specific classification functions. The mapping into the high dimensional space is performed by a function called a kernel. If the data points are images, properties like spatial correlation can be used to derive kernels which perform a "good" mapping and so induce a better classification. I investigate suitable kernels for images and show how a variant of the similarity measure described above is used with success in a number of applications.

Novelty detection (i.e., learning one class at a time): in many applications, while it is relatively easy to represent the class of objects one wishes to identify, it is not straightforward to understand what represents the opposite class (in face detection problems, for instance, it is easy to collect examples of faces, while it is not easy to decide what the non-faces are). Representing and classifying one class at a time is one possible solution. In this framework I have studied and tested extensively the kernel for images introduced.

Each contribution introduced in this thesis to the above problems is somewhat independent from the others and can be used to solve specific applications but, taken all together, they can represent the main tools for a complex object representation and identification system.

SO LONG AND THANKS FOR ALL THE FISH
(*Douglas Adams*)[1]

[1]Pan Books, London, 1984.

A Tiziana — prima perché c’era, ora perché non c’è

Acknowledgements

There is a person — who I am not going to name [Isg01], who wrote the best lines of acknowledgement I've read so far: not a name or a detail, but they couldn't be clearer. After that, everything seems banal, but still...

Very special thanks to some of the people I've had the pleasure to work with and to learn from: Alessandro Verri, who first suggested to me this journey and since then didn't stop supporting me, Emanuele Trucco, who gave me the chance to work with him and to see the world, Andrea Fusiello and Francesco Isgrò who, in different ways, taught me about curiosity and enthusiasm for research.
To all the people (too many to list them all) I shared lab 304, lab G78, room 309, and Vicolab with: thanks for all the inspiration, the support, the chats, the laughs, and the coffee breaks...

Since life is not just work, here there are a few other reasons why these years have been special: the `Informatics Angels` — Anna, Ba, Cazza, Michi, Rimi, Rocca: vi voglio bene. The USA2001 team and all the Azzarelli family: fantastic!
The family of Brougham Place — Charlie, Tony, Brenda and all their horses — and all the other people passing by — Tiziana, Andrea, Julo, Corinna, Peter, Gianluca, ...
Stefano for all that stuff about friendship that now I know it's true, and for a bunch of other things like trust, patience, laughs, and a mobile phone.
Alberto, for a friendship which, as the time passes, is never the same and it is still challenging, and Fro for ofr rfo orf rof.

...TIMOTHY! (words just can't tell...)

Ed infine, grazie a mamma e papà per quello che siete — e quello che sono!

Canneto, 8 Novembre 2001

Table of Contents

Introduction

Scope and motivation

This thesis considers the representation and the identification of 3D objects in image sequences: objects are represented by information extracted from image sequences, and this representation is exploited to solve problems such as object detection, recognition, and scene location.

For these purposes the work of this thesis has taken three main directions, which can be summarized as follows: the analysis of image sequences, from the point of view of the 2D motion information they carry; the extraction of meaningful information from image sequences, and the modeling of a 3D object as a simple collection of multiple views; the use of this low-level representation in a statistical learning framework, in order to produce a higher level object representation for detecting or recognizing instances of the object in single novel views.

Two important characteristics of our approach are that we use unconstrained image sequences acquired by a non calibrated camera, and that we aim at limiting the preprocessing of the images, in particular the computation of images correlations. Our idea is based on finding one possible collection of characterizing views of an object which describes a low-level representation; once this first representation stage is completed, the images are represented as vectors, and are given to a learning machine that uses a suitable kernel for images to learn one class at a time, that implicitly produce a high level representation of the object. The kernel for images we devised is tolerant to small spatial misalignments, therefore reducing the need for image correlation.

The idea of using image sequences as a basis to perform object representation is not new. 3D object representation or modelling is a central problem of various disciplines, ranging from photogrammetry (modelling for measuring), through computer graphics (modelling for visualization or interaction in virtual environments), computer vision (modeling for analysis), to the more recent computer animation, not to mention the various hybrid applications which lie among them. In the last decades many ideas have been proposed, and among them, there are a few that use image collections or image sequences as input data.

The first family of methods, which originates in computer graphics and engineering, is based on an explicit definition and construction of *full 3D models*. Since this approach is the most distant from ours in the range of existing methods we will not discuss it further (for a review, the reader may refer to [BJ85]; a more specific work that addresses 3D object contruction from image sequences, is [Pol]). Usually such models are sophisticated, difficult to build, and often hard to use. *Aspect graphs* [KvD79, BJ85], instead, are one of the first attempts to represent the 3D appearance of objects in terms of their 2D projections; with aspect graphs, though, there is no computational gain relative to computing the full 3D model of the object. However, the idea of representing objects using 2D rather than 3D models is still popular: it has been supported by recent advances in biological vision [BET95, Sin95], and has been adopted by various researchers. Among other techniques, it is worth mentioning 2D *morphable models* [JP98, PV92, VP97, BSP93] and *view-based active appearance models* [ECT99, CWT00], which use a selection of 2D views for the purpose of modeling a 3D complex object. The idea behind both methods (which achieve comparable results in two rather different ways) is that of synthesizing a model for each object or class of objects, and then matching it with a novel image to check for consistency. Both morphable models and active appearance models are firmly based on registering a selection of images belonging to the same object or the same class, and compute linear combination of the examples in terms of their 2D shape and their texture, which represent a model of the object. For fixed values of the model parameters, the model can be rendered as a valid image of the object represented. Their images belong to a linear space of representations for each object or class — that is, a linear combination from a selection

of faces is still a face, thanks to registration, hence, these methods model a linear space of the faces. For this reason, given a model of an object or a class, they can also be used to generate new synthetic views.

Our approach, instead, starts from an image sequence and first finds one of the possible collections of representative views of an object; it then represents these views in a space which is not necessarily linear. Unlike aspect graphs, this representation is not unique, and can vary considerably depending on imaging conditions, image sequence quality, etc. With respect to morphable models and active appearance models, our approach gains in simplicity in the representation stage, while it loses the ability to automatically generate synthetic views — our representation space is not necessarily linear since our images have not been registered to start with.

The *learning from examples* paradigm has been applied with success to object identification and recognition. Strongly inspired by the learning capabilities of young humans, it is based on describing a phenomenon, an object, or an action by means of examples. The classical approach that allows the system to be able to discriminate correcly and minimizes the number of incorrect answers to new questions is to show the system both positive examples (examples which correspond to the object or action we are describing) and negative examples — in the case of image-based learning the examples will consist of images or portion of images. While positive examples are usually defined as images containing the object of interest, negative examples are comparatively much less expensive to collect, but equally somewhat ill defined and difficult to characterize as a class. Almost all researchers in this field seem to agree on the fact that a representative list of informative negative examples should be obtained by carefully selecting the most difficult negative examples (i.e., the negative examples which are most likely to be mistaken as positive) [PP00]. An alternative to this approach, however, is to design a learning system that learns from a class of representative positive examples only. It is beyond doubt that this approach, in the best case, can perform as well as the standard positive-negative (binary classification) approach, but that in general it may perform worse, since it uses less information; on the other hand, we are encouraged by our results, and by the fact that the time and effort needed for data collection decrease enormously.

Thesis structure

This thesis is divided into three parts which deal with (1) the analysis of image sequences, (2) the study of similarity measures for grey-level images, and (3) a high-level object representation and identification, based on statistical learning. Each part presents a separate aspect of the area examined, and could be applied independently to solve specific problems; together, however, they can form the main parts of a complex system for object representation and recognition. Each part opens with a chapter introducing the theoretical background necessary to the understanding of the following chapters. However, because of the variety of the topics addressed in this thesis, we have preferred to refer the reader to more appropriate references in some cases.

Part I first addresses 2D motion analysis in image sequences, and then mosaic construction and motion segmentation. It begins with an introduction to the geometry of imaging and to a formalization of the relation between corresponding points in pairs of images (Chapter 1); it continues with a description of a sparse method for 2D motion estimation based on corner tracking, and of a technique that, under certain conditions, extend the sparse representation of the 2D motion field to a global transformation of one frame into the next (Chapter 2); it concludes by describing a global sequence registration technique, which we use for mosaic construction and motion segmentation (Chapter 3). Also, as an application parallel to the main stream of this thesis, it addresses video sequence compression.

Part II introduces a similarity measure for grey-level images which can, among other applications, be used to track objects within sequences, thus extracting a collection meaningful views from an image sequence. This similarity measure is based on a metric called the Hausdorff distance. Part II starts by introducing this, and describing in which computer vision applications it has previously been used (Chapter 4). Part II next describes our similarity measure, justifies the statement that it can also be seen as a correlation method, and discusses extensions of this Hausdorff method to enable a multiscale approach, and to deal with issues of illumination variation; finally it describes how the method can be applied to solve different computer vision problems, both as a correlation measure and

as a similarity measure for the comparison of a test image to a model (Chapter 5).

Part III is devoted to describing a kernel-based method to learn one class at a time. It starts by introducing kernel methods (Chapter 6). It continues with a description of to design a kernel method for learning with positive examples only, and introduces the issue of kernel design for images. In particular it reworks the similarity method described in part II to produce a kernel for images that can be used with success in object identification. It concludes with a description of two applications, face identification and 3D object recognition (Chapter 7).

Part I

Image Sequence Analysis

Outline of Part I

Image registration[Bro92, GM99] is the process of determining correspondence between all points belonging to images of the same scene. By registering two images, information from different sources can be combined, the geometry of the scene can be recovered, and changes occurring in the scene during image acquisition can be determined. Image registration has application in fields as diverse as medical image analysis [HE98], the processing of remotely sensed data [SP97], robot vision, and industrial inspection.
In this first part of the thesis, we introduce a methodology using image registration for the compact representation of image sequences [OFT01]. The only property required of the image sequences is spatio-temporal continuity, that is, that they should be captured with a single operation of the camera and should present an action continuous in time and space; such a sequence is referred to as a *video shot*.

Chapter 1 reviews the facts about the geometry of one and two cameras; in this chapter we principally present a formalization of the geometric relation between corresponding points in a stereo pair, and discuss under which circumstances corresponding points can be related by a homography [SK52] — a non-singular linear transformation of the projective plane into itself.

Chapter 2 introduces a method to estimate a sparse 2D motion field from an image sequence, which is based on the Shi-Tomasi-Kanade feature tracker [TK91, ST93]. With this method we compute correspondences between adjacent frames, and use them to produce a global registration of the image sequence, provided that corresponding points in each

pair of images can be related by a homography.

Chapter 3 extends the registration technique described in Chapter 2 to mosaic construction. It introduces the problem of motion segmentation, and describes the mosaic-based segmentation method which we also use to accomplish video coding.

Chapter 1

Geometry of Imaging

This chapter is devoted to a review of the basic notions about one and two camera geometry for the perspective camera model, and meant as a background introduction useful for the succeeding chapters. For details on this subject the reader can refer to [TV98, HZ00].

1.1 Geometry of one camera

The most common geometric model of a camera is the *perspective* or *pin-hole* model (Figure 1.1), which consists of a plane π, the *image plane*, and a 3D point $\mathbf{O}$, the *center of projection*. The distance between π and $\mathbf{O}$, f, is the *focal length*. The straight line through $\mathbf{O}$ and perpendicular to π is the *optical axis*. The point of intersection between the optical axis and the image plane is the *principal point*. The reference frame shown in Figure 1.1 with origin in $\mathbf{O}$ and Z axis corresponding to the optical axis, is called *camera reference frame*.
The fundamental equations of the perspective model, which describe the perspective projection of a 3D point $\mathbf{P}$ in the point $\mathbf{p}$ of the image plane are:

$$\begin{aligned} x &= f\tfrac{X}{Z}, \\ y &= f\tfrac{Y}{Z}, \end{aligned} \tag{1.1}$$

In homogeneous coordinates 2-D points in the image plane can be denoted as as $\tilde{\mathbf{p}} = (x_1, x_2, x_3)$ with $\mathbf{p} = (u, v) = (x_1/x_3, x_2/x_3)$ being the corresponding Cartesian coordinates

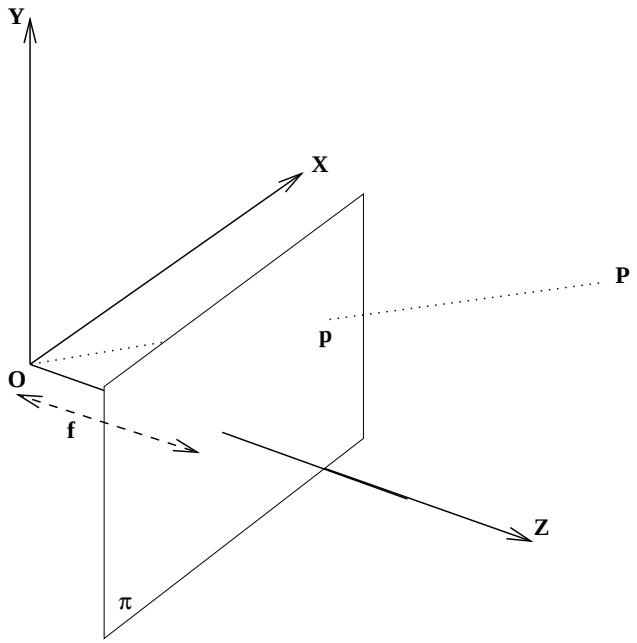


Figure 1.1: The perspective camera model

if $x_3 \neq 0$ [1]. Equations (1.1), expressed in homogeneous coordinates, become linear. Let

$$\tilde{\mathbf{P}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{p}} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix},$$

be the 3D point and its projection onto the camera image plane, respectively, then

$$\kappa\tilde{\mathbf{p}} = M_0\tilde{\mathbf{P}},$$

where

$$M_0 = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Often, the camera reference frame is unknown, thus a common problem is determining

[1] We shall use the symbol ˜ to indicate homogeneous coordinates.

the location and orientation of the camera frame with respect to some known reference frame, using only image information. Image information is described in pixel coordinates, therefore a link between pixels and millimeters is needed. To deal with this problem one performs *camera calibration*, an operation that consists on estimating two sets of parameters, known as *intrinsic parameters* and *extrinsic parameters*: the former link the pixel coordinates of an image point with the corresponding coordinates in the camera reference frame; the latter define the location and the orientation of the camera frame with respect to a known world reference frame.

More in detail, the extrinsic parameters are typically a 3D translation vector $\mathbf{T}$ and a 3×3 rotation matrix R. The relation between the coordinates of a point $\mathbf{P}$ in the world frame $\mathbf{P}_w$ and the coordinates of the same point in the camera frame $\mathbf{P}_c$ is:

$$\mathbf{P}_c = [R; \mathbf{T}]\tilde{\mathbf{P}}_w,$$

where $[R; \mathbf{T}]$ represents a 3×4 matrix the first three columns of which are occupied by R and the fourth by vector $\mathbf{T}$.

Intrinsic parameters, instead, specify respectively: the perspective projection (i.e., the focal length f), the transformation between camera coordinates (mm) and image coordinates (pixel), the geometric distorsion induced by optics. The relation between the pixel coordinates $\mathbf{p}_{im}$ and the camera coordinates $\mathbf{P}_c$ of a $3D$ point $\mathbf{P}$ is:

$$\kappa\tilde{\mathbf{p}}_{im} = A\mathbf{P}_c,$$

where A is the matrix of the instrinsic parameters. Neglecting geometric distorsions, and assuming that the pixel is rectangular, considering the reference frames of Figure 1.2, A can be written as follows:

$$A = \begin{bmatrix} -f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix},$$

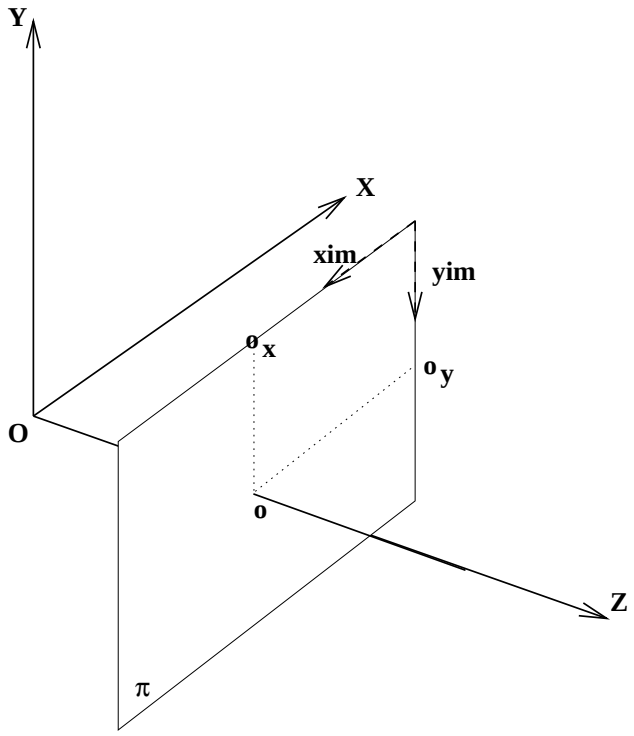


Figure 1.2: Camera reference frame and image reference frame

where f is the focal length, s_x, s_y the size of a pixel in millimeters, (o_x, o_y) the pixel coordinates of the principal point o.

In conclusion, under the assumption of a perspective camera model, the *projection matrix* that links a 3D point in world frame coordinates, $\mathbf{P}_w$ and its projection on the image plane in pixel coordinates $\mathbf{p}_{im}$, is a 3×4 matrix, so that

$$\kappa \tilde{\mathbf{p}}_{im} = M \tilde{\mathbf{P}}_w, \tag{1.2}$$

where $M = A[R; \mathbf{T}]$.

1.2 Two view geometry

The geometry of a two view system is known as *epipolar geometry*, shown in Figure 1.3. The reference frames of the left and right camera are identified by their image planes and the optical centers, O and O′ respectively. Their relative position is defined by a

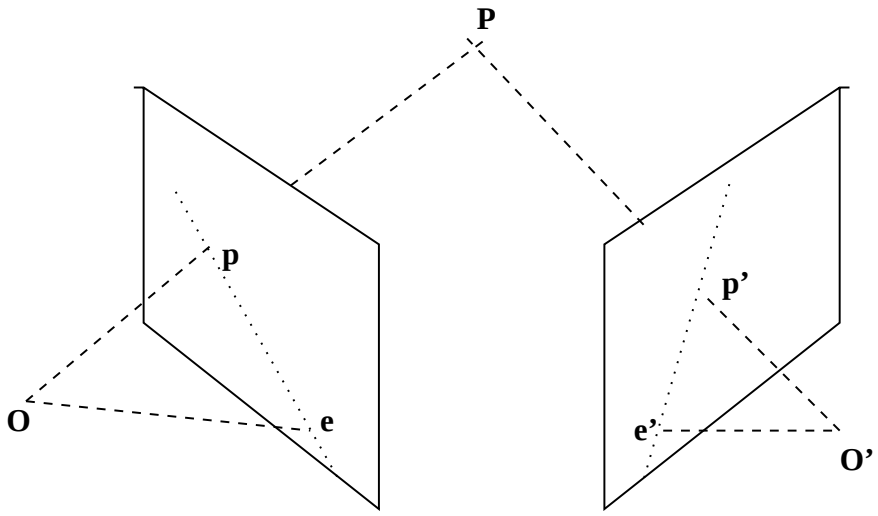


Figure 1.3: The epipolar geometry

rigid transformation of the 3D space, described by a translation vector $\mathbf{T} = (\mathbf{O}' - \mathbf{O})$ and a rotation matrix R. Given a 3D point, the relation between its coordinates in the two reference frames is:

$$\mathbf{P}' = R(\mathbf{P} - \mathbf{T}). \tag{1.3}$$

The name epipolar geometry derives from the *epipoles*, the two points $\mathbf{e}$ and $\mathbf{e}'$ which lie at the intersections between the line $\mathbf{OO}'$ and the two image planes. Any point $\mathbf{P}$ defines a plane $\pi_{\mathbf{P}}$ called *epipolar plane* which goes through the two optical centers and the point $\mathbf{P}$. It intersects the image planes in two *conjugate epipolar lines* which contain both the epipole and the projection of $\mathbf{P}$ on the image plane.

Epipolar constraint. Corresponding points must lie on conjugate epipolar lines.

Essential matrix

The equation of $\pi_{\mathbf{P}}$ can be written as the coplanarity condition of $\mathbf{P}$, $\mathbf{T}$ and $\mathbf{P} - \mathbf{T}$:

$$(\mathbf{P} - \mathbf{T})^\top \mathbf{T} \times \mathbf{P} = 0.$$

Using (1.3) we obtain

$$(R^\top \mathbf{P}')^\top \mathbf{T} \times \mathbf{P} = 0. \tag{1.4}$$

Recalling that a vector product can by written as a multiplication by a rank deficent matrix, we can write

$$\mathbf{T} \times \mathbf{P} = S\mathbf{P} \quad \text{where} \quad S = \begin{bmatrix} 0 & -T_z & T_y \\ T_z & 0 & -T_x \\ -T_y & T_x & 0 \end{bmatrix}.$$

Then, Equation (1.4) becomes

$$\mathbf{P'}^{\top} E \mathbf{P} = 0, \quad \text{where} \quad E = RS \tag{1.5}$$

is called *essential matrix*. Observe that (1.5) can be rewritten as

$$(\tilde{\mathbf{p}}')^{\top} E \tilde{\mathbf{p}} = 0, \tag{1.6}$$

which is the algebraic translation of the epipolar constraint.

Computing E: the eight-point algorithm

This algorithm is due to Longuett-Higgins [LH81]: Assume to have n point correspondences on two image planes, expressed in the camera reference frames. Each correspondence gives one homogeneous linear equation like the one of Equation (1.6); all these equations form a homogeneous linear system $A\mathbf{u} = 0$, where the unknown $\mathbf{u}$ are the 9 entries of E. If $n \geq 8$, and the n points do not form any degenerate configuration [LH81], the entries of E can be determined as the nontrivial solution of the system, and since the system is homogeneous the solution is unique up to a scale factor. The eight-point algorithm has the appeal of being simple but effective. The essential matrix is of great importance in the study of epipolar geometry, since it represents a mapping between image points in one image plane and the corresponding epipolar line in the other image plane.

The essential matrix is useful when the intrinsic parameters of both cameras are known. In the case of lack of prior information about the stereo system one can estimate the fundamental matrix, which is defined in terms of pixel coordinates. For more information on

epipolar geometry see [TV98] or [HZ00].

Relation between corresponding points

If we take the first camera reference frame as the world reference frame, we can write the two following general projection matrices:

$$M = A[I;\mathbf{0}] = [A;\mathbf{0}] \tag{1.7}$$

$$M' = A'[R;\mathbf{T}] \tag{1.8}$$

Then, for the first camera,

$$\kappa\tilde{\mathbf{p}} = M\tilde{\mathbf{P}}, \tag{1.9}$$

where κ is the depth of $\mathbf{P}$, that is, its distance from the focal plane of the first camera. Similarly, for the second camera we can write:

$$\kappa'\tilde{\mathbf{p}}' = M'\tilde{\mathbf{P}}. \tag{1.10}$$

From (1.9) and (1.8) we obtain:

$$\kappa'\tilde{\mathbf{p}}' = A'[R;\mathbf{T}]\tilde{\mathbf{P}} = A'[R;\mathbf{T}]\left(\begin{bmatrix} x \\ y \\ z \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}\right) = A'R\begin{bmatrix} x \\ y \\ z \end{bmatrix} + A'\mathbf{T}, \tag{1.11}$$

and from (1.10) and (1.7) we obtain:

$$\kappa A^{-1}\tilde{\mathbf{p}} = [I;\mathbf{0}]\ \tilde{\mathbf{p}} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{1.12}$$

Substituting the latter in (1.11) yields

$$\kappa' \tilde{\mathbf{p}}' = \kappa A' R A^{-1} \tilde{\mathbf{p}} + A' \mathbf{T}, \tag{1.13}$$

which models the relation between $\mathbf{p}$ and $\mathbf{p}'$, the projections of a 3D point $\mathbf{P}$ on the left and the right image plane, respectively.

1.3 Approximating the geometry of two cameras with homographies

A non-singular linear transformation of the projective plane [HZ00] into itself is called *homography* (or *collineation*). The most general homography is represented by a non-singular 3×3 matrix H:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} H_{1,1} & H_{1,2} & H_{1,3} \\ H_{2,1} & H_{2,2} & H_{2,3} \\ H_{3,1} & H_{3,2} & H_{3,3} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}. \tag{1.14}$$

The matrix H has 8 degrees of freedom, being defined up to a scale factor. The transformation is linear in projective (or homogeneous) coordinates, but it is *non linear* in Cartesian coordinates:

$$\begin{cases} u' = \dfrac{H_{1,1}u + H_{1,2}v + H_{1,3}}{H_{3,1}u + H_{3,2}v + H_{3,3}} \\ v' = \dfrac{H_{2,1}u + H_{2,2}v + H_{2,3}}{H_{3,1}u + H_{3,2}v + H_{3,3}} \end{cases}, \tag{1.15}$$

where $u' = x'/z'$ and $v' = y'/z'$ if $z' \neq 0$.

Two images taken by a moving camera are related by a projective plane transformation in two cases:

- the scene imaged from different points of views is planar,
- the 3D scene is viewed from the same point of view (the camera is rotating around its optical centre).

If camera is rotating ($\mathbf{t} = 0$), Equation (1.13) becomes:

$$\frac{\kappa'}{\kappa}\tilde{\mathbf{p}}' = A'RA^{-1}\tilde{\mathbf{p}} \tag{1.16}$$

The 3×3 matrix $H_\infty = A'RA^{-1}$ represents a homography, and does not depend on the 3D structure.

In the other case, if the camera undergoes a general rigid motion, but 3D points lie on a plane Π with Cartesian equation $\mathbf{n}^\top\mathbf{P} = d$, Equation (1.13) can be specialized, obtaining:

$$\frac{\kappa'}{\kappa}\tilde{\mathbf{p}}' = \left(H_\infty + \frac{A'\mathbf{T}\ \mathbf{n}^\top A^{-1}}{d}\right)\tilde{\mathbf{p}}. \tag{1.17}$$

Therefore, there is a projective plane transformation between the two views induced by the plane Π, given by $H_\Pi = H_\infty + A'\mathbf{T}\frac{\mathbf{n}^\top}{d}A^{-1}$. The H_∞ homography, obtained in the previous case, can be interpreted as the homography induced by a very special plane, *the infinity plane*, as can be seen by letting $d \to \infty$ in (1.17).

It might be worth showing how two views are related in the general case of full 3D scene and arbitrary camera motion. Starting again from Equation (1.13)

$$\kappa'\tilde{\mathbf{p}}' = \kappa H_\infty\tilde{\mathbf{p}} + A'\mathbf{T}, \tag{1.18}$$

and substituting $H_\infty = H_\Pi - A'\mathbf{t}\frac{\mathbf{n}^\top}{d}A^{-1}$, we obtain (from Eq. 1.17):

$$\frac{\kappa'}{\kappa}\tilde{\mathbf{p}}' = H_\Pi\tilde{\mathbf{p}} + A'\mathbf{T}\left(\frac{a}{d\ \kappa}\right). \tag{1.19}$$

where $a = d - \mathbf{n}^\top\kappa A^{-1}\tilde{\mathbf{p}}$ is the orthogonal distance of the 3D point $\mathbf{P}$ (of which $\mathbf{p}$ and $\mathbf{p}'$ are projections) to the plane Π.

If $\mathbf{P}$ is on the 3D plane Π, then $\tilde{\mathbf{p}}' \simeq H_\Pi\tilde{\mathbf{p}}$. Otherwise, the remaining displacement, called *parallax*, is proportional to the *relative affine structure* $\gamma = a/(d\ \kappa)$ of $\mathbf{P}$ (wrt the plane Π) [SN96]. The relative affine structure of a point depends on its depth, on the choice of the first view and on the reference plane. When the reference plane is the plane at infinity, the relative affine structure reduces to $\gamma = 1/\kappa$, as can easily be seen from Eq.(1.13).

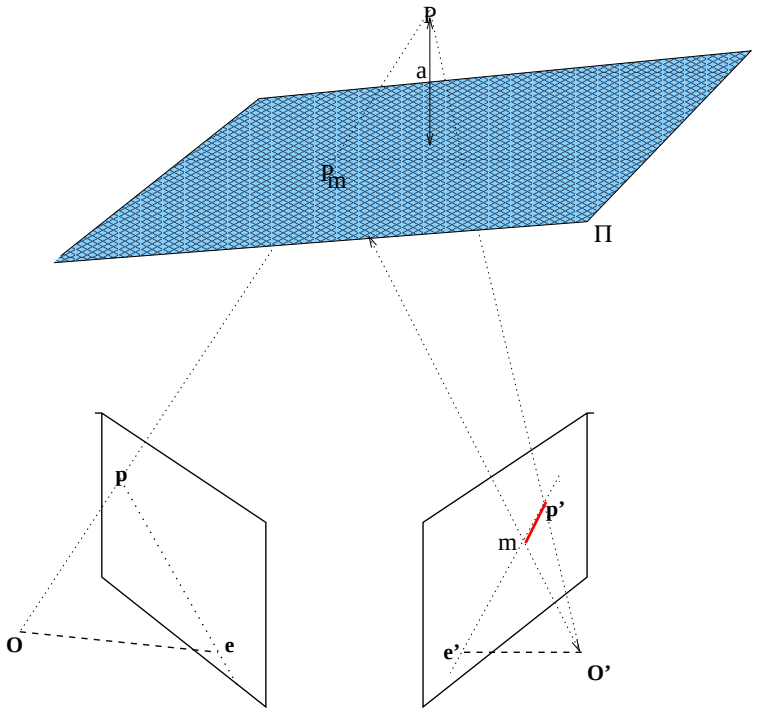


Figure 1.4: Error made in assuming that the scene is planar when it is not. The segment $\mathbf{mp}'$ is caused by the assumption that the scene is planar and therefore that the 3D point which originated $\mathbf{p}$ is not $\mathbf{P}$ but it is $\mathbf{P_m}$.

Sometimes, if the effect of the parallax is neglectable, it is convenient to ignore it and estimate the relation between corresponding points in two images by estimating a homography. Figure 1.4 shows the error made in the case of such an assumption. In the next section we will see how a homography can be estimated from point correspondences.

1.4 Homography computation

Since H has 8 d.o.f., eight independent parameters are required to define the homography. Each correspondence (u, v), (u', v') provides two equations:

$$\begin{cases} u'(H_{3,1}u+H_{3,2}v+H_{3,3}) = H_{1,1}u+H_{1,2}v+H_{1,3} \\ v'(H_{3,1}u+H_{3,2}v+H_{3,3}) = H_{2,1}u+H_{2,2}v+H_{2,3} \end{cases} . \tag{1.20}$$

We conclude that four points (provided that no three of them are collinear) determine a unique homography H.

There are two methods of dealing with the unknown scale factor in a homogeneous matrix: to fix the value on one of the matrix elements, usually $H_{3,3} = 1$, or to solve for the matrix up to a scale. We used the latter, which is more general. Equation (1.20) can be rearranged as:

$$\begin{bmatrix} u & v & 1 & 0 & 0 & 0 & -uu' & -vu' & -u \\ 0 & 0 & 0 & u & v & 1 & -uv' & -vv' & -v' \end{bmatrix} \begin{bmatrix} H_{1,1} \\ H_{1,2} \\ H_{1,3} \\ H_{2,1} \\ H_{2,2} \\ H_{2,3} \\ H_{3,1} \\ H_{3,2} \\ H_{3,3} \end{bmatrix} = \mathbf{0}. \tag{1.21}$$

For $n \geq 4$ points $\mathbf{p}_1 = (u_1, v_1)^\top \ldots \mathbf{p}_n = (u_n, v_n)^\top$, we obtain a rank deficient system of

homogeneous linear equations, which has the form $L\mathbf{h} = \mathbf{0}$, where

$$L = \begin{bmatrix} u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1u'_1 & -v_1u'_1 & -u_1 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & -u_1v'_1 & -v_1v'_1 & -v'_1 \\ . & & & & & & & & \\ . & & & & & & & & \\ . & & & & & & & & \\ u_n & v_n & 1 & 0 & 0 & 0 & -u_nu'_n & -v_nu'_n & -u_n \\ 0 & 0 & 0 & u_n & v_n & 1 & -u_nv'_n & -v_nv'_n & -v'_n \end{bmatrix},$$

and $\mathbf{h} = [H_{1,1}, H_{1,2}, H_{1,3}, H_{2,1}, H_{2,2}, H_{2,3}, H_{3,1}, H_{3,2}, H_{3,3}]^\top$. If $n > 4$ there are more equations than unknown, and, in general, only a Least Squares solution can be found. Usually it is advisable to use all the correspondences, to prevent inaccuracies caused by noise or feature misplacement.

One Least Squares solution is the column of V corresponding to the least singular value of L, where $L = UDV^\top$ is the Singular Value Decomposition (SVD) [GL96] of L. The computational cost of SVD is $O(n^3)$.

As pointed out by Hartley in the case of the fundamental matrix estimation, a better conditioned problem is obtained by data *standardization*[Har95]. The points are translated so that their centroid is at the origin and are then scaled so that the average distance from the origin is equal to $\sqrt{2}$. Let T and T' the resulting transformation in the two images and $\tilde{\mathbf{p}}^* = T\tilde{\mathbf{p}}$, $\tilde{\mathbf{p}}'^* = T'\tilde{\mathbf{p}}'$ the transformed points. Using $\tilde{\mathbf{p}}^*$ and $\tilde{\mathbf{p}}'^*$ in the homography estimation algorithm, we obtain a matrix H^* that is related to the original one by $H^* = T'HT^{-1}$, as it can be easily seen.

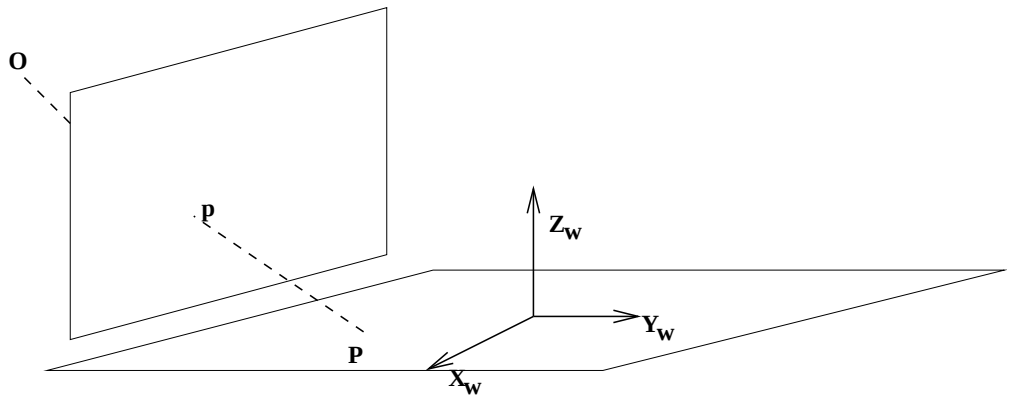


Figure 1.5: The map between a world plane and the perspective plane is a homography

1.5 Metric rectification

According to Equation (1.2), the map between a 3D point $\mathbf{P}$ and its projection onto the image plane is given by a 3×4 matrix (in homogeneous coordinates) such that:

$$\kappa \tilde{\mathbf{p}} = M \tilde{\mathbf{P}}, \tag{1.22}$$

where κ is the depth of $\mathbf{P}$.

The map between a world plane not passing through the optical centre and its corresponding perspective image is a homography: indeed, if we choose the world coordinate system such that the points on the world plane have Z coordinate equal to zero, then the perspective projection matrix M reduces to a full rank 3×3 matrix representing a general plane to plane projective transformation. In fact, we have that

$$\kappa \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} M_{1,1} & M_{1,2} & M_{1,3} & M_{1,4} \\ M_{2,1} & M_{2,2} & M_{2,3} & M_{2,4} \\ M_{3,1} & M_{3,2} & M_{3,3} & M_{3,4} \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} M_{1,1} & M_{1,2} & M_{1,4} \\ M_{2,1} & M_{2,2} & M_{2,4} \\ M_{3,1} & M_{3,2} & M_{3,4} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{1.23}$$

where the 3×3 matrix is full rank, as otherwise the projection of the plane would degenerate to a line.

A homography is fully defined by four points of which we know the relative position in the world plane. Once the homography is determined, the image can be back projected onto the object plane. This is equivalent to synthesize an image from a fronto-parallel view of the plane. This is known as metric rectification [LZ98] of a perspective image.

Chapter 2

2D motion estimation and sequence registration

The main low level operation of image sequence registration is 2D motion estimation. In this chapter, we first introduce a motion estimation method based on Shi-Tomasi-Kanade feature tracking; we then show how, under some assumptions, the estimated motion field can be generalized to a global geometric transformation (more precisely, a homography) of one image of the sequence into another. As we anticipated in Chapter 1, this happens if the camera performs a pure rotation (as in a panning operation), or if the scene can be well approximated by a single plane (that is, the depth range of the scene is small compared to the distance from the camera). We describe a robust technique *for estimating the homographies corresponding to the dominant inter-frame motion, which deals with possible moving objects in the observed scene by treating the points belonging to moving objects as outliers. By composing all the different homographies between adjacent frames, it is possible to obtain the transformations relating each image of the sequence to an arbitrarily chosen reference frame.*

2.1 Introduction and comparison with previous work

In this section we introduce our approach to the registration of image sequences, and review some other approaches presented in literature, mainly from the point of view of

the applications discussed in Chapter 3 — mosaic construction and motion segmentation. Our approach to 2D motion estimation is based on the so called *feature-based* methods. It uses feature tracking to collect sparse correspondence points across an image sequence, and then obviates the sparsity of the resulting representation by estimating a global transformation of one image into the next: by employing a statistically robust method here, the points belonging to (possibly multiple) objects in motion are treated as outliers, assuming that the majority of points follows a common motion. We achieve sequence registration, by composing these global transformations to compute the homography relating each image to a common reference frame.

Sparse approximations are used whenever sparse but reliable results are sufficient for the task and a low computational complexity is required. In mosaicing and image alignment, it seems most natural to look for a transformation of all the pixels from one frame to the next, hence dense motion field approximations are common [GJ98, HAD+94, IAH95, RPFRA98, SA96, Sze96], while there seem to be few attempts to use feature-based techniques [ZFD97, DC95], which are, however, popular in other motion analysis applications.

One of the most common choices for image alignment is the direct minimization of discrepancy in pixel intensities[IAB+96, SA96, Sze96]. This technique is closely related to computing a dense approximation of the 2D motion field, that is, the apparent motion of the image brightness pattern (the optical flow) [BFB94, CV92, HS81, TV98].

Zoghlamy *et al.* [ZFD97] developed an interesting corner-based method for 2D mosaic construction. Their main concern was to obtain the best possible homography, and their first approach was therefore to compute all the possible homographies obtainable from pairs of corner fourtuples, and then to select the best (i.e., the one which maximizes a similarity function over all the corners). The results were very accurate but the technique is too computationally expensive. They subsequently limited the number of possible corner combinations by adopting a particular *corner model* [DB93].

Our method is based instead upon using all the correspondences available between pairs of images to compute only one homography, and choosing a statistically robust technique to limit the influence of outliers. We use motion information only where it is most reliable,

i.e., derived from points which do not suffer from the aperture problem, thus keeping the computational complexity low (e.g., by controlling the number of features). We then estimate the discrepancy for each pixel, by computing a global 2D motion model for the whole image, using the reliable motion information at feature points.

Whereas optical-flow techniques are dense both in time and in space, standard feature-based techniques are sparse in space and also in time, since they typically use frames with a moderate overlap and rely on feature matching. Instead, because we use feature *tracking*, our approach is sparse in space but dense in time. This makes the feature matching fast and reliable, as the features do not change too much from one frame to the next, and the estimation of their motion is unambiguous since they do not suffer from the aperture problem.

2.2 Computation of sparse correspondences

In this section we briefly describe a sparse 2D motion estimation method, upon which the Shi-Tomasi-Kanade tracker is based, that we use to obtain sparse correspondences across the image sequence. For more details we refer the reader to the seminal works by Shi and Tomasi [ST94] and Tomasi and Kanade [TK91], and, for a statistically robust extension of the method, to Fusiello *et al.* [FTTR99].

2.2.1 Feature tracking

Consider an image sequence, and a set of points (features) of the image plane, $\{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$, at instant t. Feature tracking consists of estimating the position of the corresponding points in the image plane acquired at instant $t + 1$.

Let $\mathbf{p} = (x, y)^\top$ be an image point, the intensity value of which at time t is $I(\mathbf{p}, t)$. It is common experience that, under most circumstances, variation of the apparent brightness of objects is not significant. Thus, if the time sampling frequency is sufficiently high, we can assume the constancy of the apparent brightness of the observed scene, that is, that the image intensity of any given point in the scene is stationary with respect to time:

$$\frac{dI}{dt} = 0. \tag{2.1}$$

Since the intensity depends upon space and time, we rewrite Equation (2.1) as:

$$\frac{dI(x(t), y(t), t)}{dt} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0, \tag{2.2}$$

which leads to:

$$(\nabla I)^\top \mathbf{v} + I_t = 0, \tag{2.3}$$

where $\nabla I = (I_x, I_y)^\top = (\partial I/\partial x \ \partial I/\partial y)^\top$ is the spatial image gradient, $I_t = \partial I/\partial t$, and $\mathbf{v} = (v_x, v_y)^\top = (dx/dt, dy/dt)^\top$ is the motion field.

Equation (2.3), which is known as *image brightness constancy equation*, is not sufficient to estimate the motion field; it allows the estimation of only the component of the motion field in the direction of the image gradient (the so-called *normal component*), as described in [TV98]. This problem is known as *aperture problem*, and can be explained by saying that the component of the motion field that is orthogonal to the image spatial gradient is not constrained by the image brightness constancy equation.

One practical and simple way to estimate the motion field is to start from the assumption that the motion field is well approximated by a vector field, $\mathbf{v}$, which is *constant* within small regions of the image plane. This means that, given a point $\mathbf{p}$ of the image plane, for each point $\mathbf{q}$ within a small image window N_p centered at $\mathbf{p}$ we can write: $(\nabla I)^\top \mathbf{v} + I_t = 0$. As the image motion model is not perfect, and because of image noise, Equation (2.3) is not satisfied exactly; the problem now is to find the vector $\overline{\mathbf{v}}$ which minimizes the functional:

$$\Psi(\mathbf{v}) = \sum_{q \in N_p} \left[(\nabla I)^\top \mathbf{v} + I_t\right]^2. \tag{2.4}$$

The solution to this least squares problem can be found by solving the linear system

$$G\mathbf{v} = \mathbf{e}, \tag{2.5}$$

where

$$G = A^\top A = \sum_{\mathbf{q}\in N_p} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}, \tag{2.6}$$

$$\mathbf{e} = A^\top \mathbf{b} = -\sum_{\mathbf{q}\in N_p} \nabla I(\mathbf{q})^\top I_t(\mathbf{q}). \tag{2.7}$$

2.2.2 Feature extraction

The motion estimation technique described above can be applied for both feature tracking (in which case it is preceeded by a feature extraction phase, and applied only to a selected set of points — the features), and dense estimation of the motion field (in which case it is applied at all image points). Feature tracking, which we prefer here, is based on selecting identifiable points from the sequence, and therefore estimates the motion field only where it is most reliable. For an application of the motion estimation method to the dense case, the reader may refer to [TV98].

Our features are points that are "good to track" [ST94], in that they are unambiguously detectable from one frame to the next. This is equivalent to requiring that we can find a numerically stable solution to Equation (2.5), i.e., that G be well-conditioned and have entries which are well above the noise level. Indeed, let us consider the matrix G defined by Equation(2.6). We note that G is symmetric and so can be diagonalized by a rotation of the coordinate axis, and thus, without loss of generality, we can take G to be a diagonal matrix:

$$G = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}.$$

Both eigenvalues λ_1 and λ_2 are non-negative. If we assume $\lambda_1 \geq \lambda_2$, the points for which G is numerically stable are points where $\lambda_1 \geq \lambda_2 > 0$. Such points are the so-called *corners* — image points characterized by an intensity discontinuity in two directions.

Starting from G we can classify the image points according to the amount of information that their neighbourhood carries: if the image gradient vanishes everywhere (the intensity

pattern is uniform) $\lambda_1 = \lambda_2 = 0$; in the presence of a strong (high contrast) edge $\lambda_1 > 0$ while $\lambda_2 = 0$ instead; finally, in the presence of a corner, we expect $\lambda_1 \geq \lambda_2 > 0$. The larger the eigenvalues, the stronger the corresponding image lines.

In practice, to extract corners from an image, for each image point $\mathbf{p}$ we compute matrix G with respect to a neightbourhood N_p, and then find its eigenvalues; the requirement for point $\mathbf{p}$ to be a corner is that its smaller eigenvalue is sufficiently large: we accept the point if

$$\min(\lambda_1, \lambda_2) > \lambda,$$

where λ is a user-defined threshold [ST94].

Corners possess the curious property that they do not necessarily correspond to geometric features of the real world, but rather capture corner structures in the image pattern. These are intersections of image lines which correspond to boundaries in the real scene (of solid objects or of shadows, etc.) — these, however, may lie at different depths in the 3D world.

2.3 Estimation of the dominant motion

In this section we shall see how to use homographies to approximate the transformations between pairs of images, and how to cope with the presence of moving objects.

Let us suppose that we are given an image sequence with negligible parallax (i.e., adjacent frames are approximately related by a homography) and that point correspondences across the image sequence have been obtained by feature tracking. These correspondences can be used to compute the transformation matrix for each pair of adjacent frames. It is known (as discussed in Section 1.4) that four point correspondences are sufficient to compute the homography matrix uniquely (up to a scale factor).

If the correspondences have been obtained by feature tracking, there may be a large number of pairs available. If the scene is static and the motion is induced by the camera motion only, then all features will represent the same motion; if they are all included in the homogeneous system, the Least Squares estimate is usually accurate enough, even in the presence of a small number of outliers due to the feature tracker errors.

In the presence of moving objects the number outliers increases, since each feature of a moving object is an outlier. In this case, therefore, a robust method must be employed in order to estimate the homography that explains the motion of the *majority* of the features, that is the *dominant motion* [IRP94a]. Unless the scene is cluttered with many moving objects, this is usually the motion of the camera with respect to the static background.

Least Median of Squares [RL87] is a robust regression technique which is very popular among the computer vision community [MMRK91, Zha97]. The principle behind LMedS is the following: given a regression problem, where d is the minimum number of points which determine a solution (four, in our case), compute a candidate model based on a randomly chosen d-tuple from the data; estimate how well the model fits to *all* the data, by means of the median of the squared residuals, where the residuals are defined, in our case, for each point correspondence, as the distances between the warped and the actual point in the second image. In formulae, let $\hat{\mathbf{H}}$ be an approximate solution of (1.21), then the residuals are

$$s_j = ||\mathbf{m}'_j - \hat{\mathbf{H}}\mathbf{m}_j|| \quad j = 1 \ldots n \tag{2.8}$$

where n is the number of point correspondences. The process is repeated on a number of data subsets, and the homography returning the smalles median square residual, MSR, is taken as the optimal model. The data points that do not belong to the optimal model that represent the majority of the data, are *outliers*. The *breakdown point*, i.e., the smallest fraction of outliers that can yield arbitrary estimate values, is 50%. In principle all the d-tuples should be evaluated; in practice, for computational efficiency, a Monte Carlo technique is applied, in which only a random sample of size m is considered. Assuming that the whole set of points may contain up to a fraction ϵ of outliers, the probability that at least one of the m d-tuple consist of d inliers is given by

$$P = 1 - (1 - (1 - \epsilon)^d)^m. \tag{2.9}$$

Hence, given d, ϵ, and the required P (close to 1), one can determine m:

$$m = \frac{\log(1-P)}{\log(1-(1-\epsilon)^d)}. \tag{2.10}$$

In our implementation we assume $\epsilon = 0.5$, and require $P = 0.99$, thus $m = 72$.

When Gaussian noise is present in addition to outliers, the relative statistical efficiency (i.e., the ratio between the lowest achievable variance for the estimated parameters and the actual variance) of the LMedS is low; to increase the efficiency, it is advisable to run a weighted LS fit after LMedS, with weights depending on the residual of the LMedS procedure [RL87].

The residuals s_j, $j = 1, \ldots, n$ are used to generate the weights for the final, weighted LS regression as follows. First, a robust standard deviation estimate [RL87] is computed as

$$\hat{\sigma} = 1.4826 \left(1 + \frac{5}{n-d}\right) \sqrt{MSR}, \tag{2.11}$$

where d is the number of parameters (4 in our case). Second, a weight is assigned to each point correspondence, such that

$$w_j = \begin{cases} 1 & \text{if } |s_j|/\hat{\sigma} \leq 2.5, \\ 0 & \text{otherwise.} \end{cases} \tag{2.12}$$

The computational cost of LMedS with Monte Carlo speed up is $O(mn \log n)$.

2.4 Sequence registration

In this section we first describe how the approximation of the 2D motion field of an image sequence arrives at a global registration of the sequence – that is, a full correspondence between each pixel of each image with pixels of the other images.

2.4.1 Registration of adjacent frames

We start by describing, from an algorithmic point of view, our feature-based 2D motion estimation technique, which consists of three steps, performed on pairs of images: *corners extraction* , *corner matching* and *homography estimation* between the images.
We perform corner extraction and matching using the feature tracker described in Section 2.2. The tracker has three main aspects:

- Extraction of features, on the first frame of the sequence,
- Tracking of features from one frame to the following,
- Re-extraction of the features, when necessary.

The re-extraction step is used when then contents of the current image have changed too much since the last extraction of corners, and thus the number of features which has survived through the tracking is too small. The re-extraction adds new features to the ones still existing.
The tracker produces a list of feature coordinates for each image. For each pair of images I_i, I_{i+1}, after all the features lost by the tracker have been discarded, we compute the homography $H_{i,i+1}$ following the procedure described in Section 2.3.

2.4.2 Global registration

The global registration of an image sequence establishes a mapping between each frame and an arbitrary reference frame. To produce the global alignment the transformation between non-contiguous frames can be obtained by multiplying the homographies of the in-between image frames. Therefore, the transformation between the image I_i and the image I_j, where $i < j$, is

$$H_{i,j} = \prod_{k=i}^{j-1} H_{k,k+1} \tag{2.13}$$

Since we have all the information necessary to proceed with global alignment, we make it consistent by choosing a reference frame and transforming each image of the sequence

with respect to it, applying Equation (2.13). The choice of the reference frame is usually guided by the application. We will see that for sequence stabilization, the sequence is usually registered with respect to the first frame, while for mosaic construction the choice is usually related to the quality of the result one want to obtain.

Once the global alignment have been completed, if we imagine to pierce all the aligned frames with a temporal line, we will intersect pixels that, in absence of parallax, correspond to the same world point, as shown in Figure 2.1.

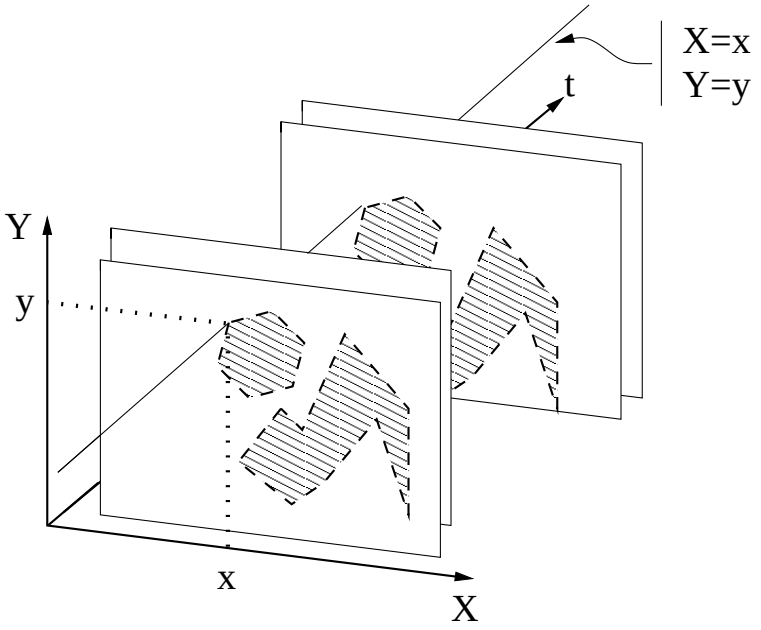


Figure 2.1: Temporal alignment: once all the frames are aligned, a temporal straight line will intersect all the frames in corresponding points.

2.4.3 Sequence stabilization

As a first instance of the use of global sequence registration we present an example of sequence stabilization. Often videos are acquired from unstable sources; *image stabilization* [DR96, IRP94b, MC96] is defined as the process of generating a compensated video sequence where image motion resulting from camera motion has been partially or totally removed. In [IRP94b], for instance, it is assumed that the translation of the camera is the intended motion (the movement of a car), while the camera rotation is the result of the

camera jittering along some rotational axis and is a cause of de-stabilization. In general, there may be, on the one hand, components of the camera motion which are meaningful, and, on the other, those which can be discarded as being due to noise, unintended or involuntary movements of the camera, or simply as uninteresting — the nature of this distinction, though, is highly dependent upon both the intended application and the particular circumstances of capture.

In our present work, all the video sequences are acquired by a hand-held camera, so the major undesirable effect is likely to be shake, which is most noticeable and disturbing when the camera is supposed to be still — unfortunately this is often precisely when we most desire a clear and noise free image.

For the purposes of this example we therefore assume that the camera is intended to be still, and the result we wish to achieve is a complete compensation for the camera motion through a suitable transformation of each image. In the stabilized image, scene points will (ideally) be motionless in spite of camera motion.

Once the global registration of the sequence has been completed, sequence stabilization is straightforward, since it consists simply of choosing a *reference image*, I_{ref} (typically the first one), and warping all the images I_j of the sequence to bring them into the same reference frame by applying the inverse of the transformation $H_{ref,j}$.

Chapter 3

Mosaic-based motion segmentation

In this chapter we examine how the sequence registration technique described in Chapter 2 can be used for mosaic construction and for mosaic-based multiple motion segmentation. We then demonstrate its successful application to address a range of problems, including video coding and editing, and, in particular, content-based video representation compatible with the MPEG-4 specifications.

3.1 Introduction

The need for a compact representation of video shots has been observed in diverse applications such as video compression [IAH95], coding [OFT00], editing [GJ98] and indexing [BCB98, BMM99, CCM+97, IHA98]. We achieve this goal as a by-product of robust 2D motion estimation: starting from our global representation of the dominant motion, we first generate a mosaic of the background, and then perform motion segmentation and produce the sequences of the foreground moving objects by comparing the background with the original sequence.

This representation of a video shot as a background mosaic together with foreground sequences is compact since all the information about the background (which does not change) is stored only once. Further, it is useful for video coding, since it achieves high compression rates in the transmission of the sequence, and meets the requirements of the MPEG-4 standard [KPC97], in which a scene is described as a composition of several *video*

objects, each one encoded separately.

A *mosaic* is a panoramic image obtained by collating all frames of a sequence or set of images after aligning (warping) all the images into a common reference frame. The result can be regarded as a *panoramic image* acquired by a virtual camera, especially useful where a single, real camera would limit resolution or could not be used at all [Sze96, TPR+00]. Besides video compression, video coding and editing, and automatic indexing of video data, mosaicing techniques are also useful for image stabilization [HAD+94] and for building high quality images with low-cost imaging equipment [CZ98].

As regards *motion segmentation*, the problem can be stated as follows: given a sequence of images, classify the pixels of each frame either as moving according to camera motion or as moving independently. In many works [CM99, SA96, GJ98, IAB+96] object segmentation is obtained by first compensating for camera motion and then considering the residual motion.

The mosaic construction and motion segmentation method that we describe is based on the robust 2D motion estimation and global registration techniques introduced in Chapter 2. After sequence alignment, we build the mosaic by blending the warped images into a single image; we assign grey-levels to each pixel of the mosaic image, taking the median among the grey-levels of overlapping pixels. In this way, moving objects are filtered out and a mosaic of the background is obtained.

We can achieve motion segmentation by thresholding the grey-level difference between the background and each frame of the sequence. The resulting binary image should represent the silhouettes of moving objects, but in practice it is noisy for several reasons: object or illumination changes, residual misalignments, interpolation errors during warping, and acquisition noise. In order to extract only relevant moving objects, we exploit temporal coherence by tracking the centroid of each moving object over the sequence.

3.2 Mosaicing

Video mosaicing has, of late, attracted increasing interest from the field of digital video processing and analysis, for applications such as automatic indexing of video data (see

[BMM99] for a recent review), video coding and video editing.

A mosaic is an image constructed from all the frames of a scene sequence which gives a panoramic view of the scene, and is an efficient way to represent the information contained in a video sequence. Since the images belonging to a sequence usually have large overlaps, a mosaic can also provide a significant space reduction.

There are many possible descriptions of a scene that can be chosen — the following classification has been proposed by Anandan et al. [IAB+96]:

- **Salient still**[MB96, MP94]. *Static mosaics* are also referred to as *salient stills* or simply as *mosaics*. They are usually built in batch mode by aligning all frames of a sequence to a reference coordinate system, which can be either user-defined or chosen automatically according to some criteria, and by then combining all the images into a single mosaic image. The only information that is difficult to capture is the changes in the scene with respect to the background. Moving objects, for instance, will disappear or will leave blurred traces inside the mosaic, according to the temporal filter used to blend the image sequence into a mosaic. Static mosaics can be extended to deal with changes of the scene or objects in motion, usually by adding information to them. These changes can either be represented independently for each frame (as we will describe in Section 3.3) as sequences of the foreground moving objects, or can be represented as additional compact layers (which are themselves mosaics) [Ade91, IAB+96]. The latter representation is also useful in the case of complex scenes with a non-negligible 3D depth [IAB+96]. Static mosaic images exploit long term temporal redundancies and large spatial correlations, and thus are efficient scene representations, ideal for video storage and retrieval, and can also be used successfully for image stabilization, video compression, and content-based layered representation of information.

- **Dynamic mosaic.** Apart from the fact that they often must be constructed in batch mode, the main limitation of static mosaics is that they cannot completely follow the dynamic aspect of a video sequence. This requires a *dynamic mosaic*, which is a sequence of evolving mosaic images, where the content of each new mosaic is

constantly updated to be coherent with information from the current frame (the initial mosaic will coincide with the first frame read) [IAH95, IAB+96, SA96]. Since dynamic mosaics can be interpreted as a sequence of mosaics, they adapt naturally to the case of multiple motions is the observed scene. For more details on this subject we suggest [IAH95].

- **Multiresolution mosaic.** Changes in image resolution can occur within a sequence if the camera zooms or translates in the direction of the optical axis. If the mosaic is built at a low resolution, it will contain less information than would have been available in the original sequence, but building the mosaic at the highest detected resolution can cause oversampling of the low resolution frames. This problem can be handled by a *multi-resolution* structure which captures information from each new frame at its highest resolution, in this way storing all the information contained in the image sequence.

More recently the so-called 3D mosaic representations have been introduced, since 2D mosaics cannot cope with scenes characterized by a complex 3D structure or large depth variations. The two main achievements are the so-called *plane plus parallax* representation, which is based on the geometry introduced in Section 1.3 and in particular on the relative affine structure, and the *layered representation* [Ade91] which is more efficient in the case of multiple moving objects *and* 3D depth variation. In this thesis we focus on 2D mosaics; for more details on 3D mosaicing techniques the reader may refer to [Ade91, BBHP92, IAH95, IAB+96].

3.2.1 Sequence alignment

In this section we deal with the problem of creating a mosaic from a sequence of images. The construction of a mosaic is accomplished in three stages: *motion estimation*, *registration* and *rendering*. Motion estimation and registration have been introduced in Chapter 2; here they are reviewed and adapted to the case of mosaic construction. Mosaic rendering will be addressed at the end of this section.

The registration or alignment of the image frames in the sequence can be performed in three ways [IAB+96]:

- **Registration based on adjacent frames (frame to frame):** the alignment parameters (the homographies, in our case) are first computed between adjacent frames of the sequence, and they are then composed to obtain the alignment of *any* frame of the sequence with a common reference frame; in this method the sequence is fully registered before the mosaic is built. One limitation of this method is that because of homographies multiplications, the numerical errors made in estimating each homography accumulate. This can produce, at the end, a significant misalignment between the first frames and the last frames added to the mosaic, especially in the case of long image sequences.

- **Frame to mosaic registration:** to limit the problem of misalignments, for each new frame one could build a temporary mosaic and compute a homography between it and the next frame.

- **Mosaic to frame registration**: if one wishes to maintain each image in its coordinate system it can be better to align the mosaic to the current frame. This technique, as the previous one, produces a sequence of temporary mosaics. Note that the transformation between the most recent mosaic and the current frame is identical to the transformation between the previous frame and the current frame.

3.2.2 Mosaic rendering

Once the images have been aligned, they can be combined, using a temporal filter, into a mosaic image. There are several temporal filters which can be used to construct the mosaic; they all work on the intensity values belonging to the temporal line of each pixel (see Figure 2.1). Among the possible choices, we mention the following:

- The *temporal average* of the intensity values, which is effective in removing temporal noise. Moving objects may leave a “ghost-like” trace in the mosaic.

- The *most recent information*, where the entire content of the most recent frame is used to update the mosaic. A variation of this method is generally used to build the dynamic mosaics described above.

- The *temporal median* of the intensity values. Moving objects whose intensity patterns are stationary for less than half of the frames tend to disappear in the resulting mosaic — effectively they are treated as outliers. The results are sharper than those obtained with the temporal average.

- The *weighted temporal average* or *weighted temporal median*, where the weights decrease with the distance of the pixel from the frame center. This scheme aims at reducing the effect of optical distortions in the original sequence, which usually affect mainly the borders of the image.

Other temporal filters have been presented in literature. For more information on the subject refer to [IAB+96] and references within.

3.2.3 The mosaic construction

In this thesis we consider static mosaics primarily; our method of dealing with moving objects will be detailed in Section 3.3, and will be based on describing each independently moving object by using a sequence containing the changes it causes with respect to the static scene represented by the mosaic of the background.
Our method of static mosaic construction involves first registering the sequence, and then blending it into a mosaic, using a frame to frame registration method. One reason for this choice, which will become clearer in the next section, is related to the choice of using a median filter: since no incremental approximation of the median operator exists, the filter must be applied to the entire sequence, once the global registration is completed.
In Figures 3.1 and 3.2 we display the result of a mosaic construction: Figure 3.1 shows six frames from the sequence “Affresco”, which contains a pan around the Cupola of the *Padova Baptistry*, in particular, a detail of *Paradise*, a masterpiece by Giusto de’ Menabuoi. Figure 3.2 is the resulting mosaic, obtained with our frame to frame sequence alignment, a

median temporal filter, and with registration performed with respect to the first reference (which is contained in the lower right part of the mosaic). The next result that we present

Figure 3.1: Six frames from the sequence "Affresco" which represents the *Paradise* of Giusto de' Menabuoi, in the Cupola of the Padova Baptistry.

Figure 3.2: The resulting mosaic from the sequence "Affresco".

shows a post of the main portal of San Lorenzo Cathedral (Genova). Figure 3.3 contains two different frames from the sequence, with the corners superimposed, while Figure 3.4 presents the resulting mosaic. Again, we used our frame to frame mosaicing method, and registered the image sequence with respect to the first frame of the sequence. Since the door is about five meters high, and the sequence has been acquired by a person with a hand-held camera standing in front of the door and tilting the camera from the bottom to the top part of the doorpost, the first frames of the sequence are fronto-parallel views of the door (Figure 3.3, left), while the angle between the image plane and the plane of the door increases as we approach the end of the sequence (Figure 3.3, right). In this case,

by registering the whole sequence with respect to the first frame, we produce a fronto-parallel mosaic of the doorpost — notice, in Figure 3.4, that the lines of the door frame are almost parallel.

Figure 3.3: Two frames from the sequence “Door” with corners superimposed.

Figure 3.4: Mosaic of the sequence “Door”.

3.3 Foreground-background segmentation

This section introduces an approach to the problem of foreground-background segmentation, based on the mosaicing technique described in Section 3.2, and on the choice of a suitable rendering function. As we noted in Section 3.2, a 2D still mosaic is not a suitable representation when the 3D depth varies too much within the observed scene or in the presence of moving objects. The approach presented in this section has been used with success as a motion segmentation technique (in the case of multiple motions); we have also used it in the case of a static scene with a 3D depth variation, where it applies with success only in very special cases.

3.3.1 The proposed method

Our approach to motion segmentation can be described as follows: we first construct a mosaic as described in Section 3.2, choosing a suitable temporal filter (such as the median or the weighted median); if the sequence rate is sufficienlty high the blending operation produces a mosaic of all the background elements, while the moving objects disappear.
We next obtain a synthetic sequence, S, of the background by warping the mosaic into each image sequence frame, I_i, using the inverse of the $H_{ref,i}$ homography. The foreground is now segmented by comparing each frame S_i of the synthetic sequence with the corresponding frame I_i of the original sequence, using a simple grey level difference and thresholding to obtain a binary map. This binary motion map contains the blobs produced by the moving objects and other smaller blobs due to misalignments, or changes in illumination and noise. We detect an object in the first frame by choosing the area of the binary map containing the largest connected region of moving pixels, and then compute its centroid. The connected component chosen in the $(i+1)$-th binary map is the closest one to the updated centroid derived from the previous step. This is an elementary form of tracking with zero-order prediction (i.e., with a constant position assumption), coupled with an elementary data association algorithm, namely the *closest neighbour* strategy [BSF88].

We also post-process the resulting maps, to improve segmentation, using the morphological operator *closure* [Ser82] — *dilation* and *erosion* in cascade — to produce more compact regions without adding noise.

3.3.2 Object segmentation

Object segmentation can be very useful in a number of applications, especially in object recognition (as we will see in the Part 3 of this thesis), when one does not want the background to affect the recognition of the object of interest in the foreground. Object segmentation is known to be difficult for single images, and is still an open problem, in spite of a great deal of work on the subject in the recent decades. One alternative possibility is to exploit the full content of an image sequence: here we consider the possible use of the foreground-background segmentation method described above, to produce a 2D layered representation. In general more complex sequence representations, like the layered or tiled representations cited in Section 3.2 are required to handle large depth variations in a scene; in some special cases, though, a 3D scene can be represented in 2D layers using the foreground-segmentation technique described above.

Let us assume that the only 3D motion is camera motion, and that the scene is composed of a quasi-planar (or distant) background, and a 3D object in the foreground. In this case, a layered representation can be achieved by segmenting the object from the background. Indeed, if we consider a 3D scene consisting of two layers (a foreground object and the background), and if the dynamic of the image sequence is entirely due to the camera motion, the 2D motion field obtained should have two main 2D motions, one corresponding to the points of the background and the other one to the points of the foreground. The difference in these 2D motions is called 2D *parallax motion* [LHP80, IA96] (see Section 1.3) as it is induced by the effects of parallax, which are due only to camera translation and 3D scene variations. If we wish to exploit the parallax effect in order to cluster the two parts of the image characterized by the two different 2D motion fields, the camera motion should have a translational component. Also, to produce a complete mosaic of the background (without the effects of occlusions, i.e., of points of the background that are

always hidden by the foreground) the depth variation of the scene should be significant. Figures 3.5, 3.6, 3.7 present results on a sequence which meets all the above mentioned requirements: in Figure 3.5 we show two frames of the image sequence with the features superimposed; Figure 3.6 illustrates the mosaic obtained by using a median filter, thanks to which the object in the foreground has been removed; finally, Figure 3.7 presents the segmentation maps obtained.

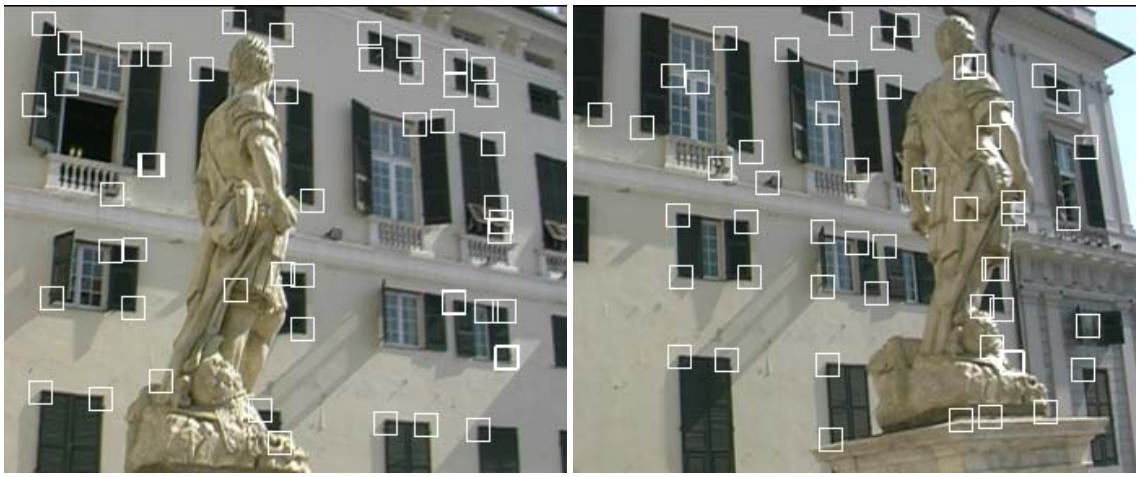


Figure 3.5: Two frames from the sequence “Statue” with corners superimposed.

3.3.3 Comparison with previous work

Other approaches to segmentation through compensation of the dominant motion have been used in the fields of surveillance, targeting, and video editing [CM99, SA96, GJ98, IAB+96]. In [SA96] motion is computed at each pixel using a robust technique, and the outlier masks obtained identify the moving object. In [GJ98] temporal analysis of grey levels is performed, based on probabilistic models and a-priori information (user-initialized), in order to segment moving objects. Irani *et al.* [IAB+96] use a local misalignment analysis based on the normal flow [IRP94a] to compare consecutive frames and extract moving objects.

However, as pointed out by Brunelli *et al.* [BMM99], such algorithms are “currently too complex to be applied to large video databases”: a low parametrical representation of the 2D motion is often sufficient.

Figure 3.6: Mosaic of the sequence “Statue”: by exploiting the effect of parallax and the fact that the background is planar, we obtain a mosaic of the background. Notice that at the center of the image there are still traces of the statue, because of the limited length of the image sequence.

Figure 3.7: Object segmentation maps, obtained by computing a thresholded difference between the synthetic sequence, S_i, and each frame of the original sequence, I_i (see text).

We require only sparse representations of the 2D motion, that possess the advantage of needing to use only reliable information from identifiable features. Also, our segmentation method, based on sparse 2D motion estimation, image differences, and blob tracking, is less computationally expensive than [SA96], and requires no user initialization (unlike [GJ98]). The reason we can use a simple thresholded image difference while other approaches (like the image flow techniques [IAB+96]) cannot, is because of the strong spatio-temporal discontinuity between the synthetic and the original sequence caused by the disappearance of the moving object: indeed, since we *first* compute the synthetic sequence that does not contain the moving objects, and *then* compare each frame of the synthetic and the original sequences, we can effectively use a simple technique such as a pixel-wise difference to identify the pixels which appear in the original sequence and not in the synthetic one — the objects in motion. In contrast, for example, the method of [IAB+96] directly compares consecutive frames of the sequence, which are naturally rather similar in continuous sequences, and thus cannot use a simple image difference. Instead, therefore, they propose *local misalignment analysis*, which is a more complex derivative-based comparison between images, and which, moreover, is not suitable for our purposes because of the spatio-temporal discontinuity mentioned above.

3.4 A sample application: MPEG4 coding

In this section we describe how the mosaic construction and motion segmentation methods explained above can be used to perform MPEG4 video compression.

MPEG-4 [KPC97] follows an approach that is called *content-based* [WA94, KPC97], based on a model of perception believed to be typical of the human brain. MPEG4 relies on a segmented representation of the video data so as to allow content-based manipulation of image sequences. A scene is considered to be composed of several Video Objects (VOs), each of which is characterized by intrinsic properties such as shape, texture, and motion. In this context, the term object has a very general interpretation and it is not necessarily a physical object — for example, the background may be considered as one VO.

A *sprite* consists of those regions of a VO that are present in the scene for some part of

the video shot (a video segment). An obvious example would be the ‘background sprite’, which could consist of a mosaic of the background in a camera-panning sequence.
The MPEG-4 standard does not prescribe a method for creating VOs, it simply provides a standard convention for describing them so that all compliant decoders will be able to extract VOs from the encoded bit-stream.
If we think of our mosaic background and the foreground sequences as VOs, the approach described in the previous section can be interpreted as a MPEG-4 compliant content-based encoding method.
In order to give a sketch of a whole video coding system, let us describe the decoder functioning. The large panoramic mosaic of the background (a sprite, in MPEG-4 terminology) is transmitted to the receiver only once. A moving foreground object is transmitted separately as an arbitrary-shape VO, described in the mosaic reference frame. Finally all the transformations between mosaic and original sequence (that is the mosaic to frame transformation) are needed. Actually it will suffice to transmit all the homographies between consecutive frames, since, starting from them, we can obtain every transformation from one reference frame to another (using Equation (2.13)). In the decoding phase, all we need do to rebuild the original sequence is to map the mosaic onto the frame of each image and paste the foreground onto it.
Content-based representation also allows for particularly straightforward editing operations on the sequence, like inserting novel Video Objects to create realistic synthetic sequences. An example of this appears at the end of this section.

3.4.1 Video compression

Figures 3.8 and 3.9 show frames from two of the sequences that we used to test the performances of our compression algorithm, which were acquired with a hand-held camera. The first sequence shown (“Super5”) is an outdoor scene with a car driving from the left to the right of the image field of view. The ego motion is nearly rotational, but a small translational component is present.
The second sequence shown (“Manuel”) has a slightly different nature: the object (person)

in motion is bigger, the natural environment under the sun produces a lot of shadows, and the depth of the scene changes significantly between the beginning of the sequence and the end.

Figures 3.10 and 3.11 show the mosaics of the backgrounds obtained with the technique explained in Section 3.2. In spite of the fact that the camera motion is not exactly rotational and the scene not planar, the registration obtained is most satisfactory in both cases. Note also that moving objects have been *automatically* removed without artifacts.

Figure 3.12 shows the results of residual analysis, performed between individual frames of the original sequence and the background (mapped onto the same frame). Figure 3.12 (left) shows the results obtained by using a thresholded difference between the 28-th frame of the sequence "Manuel" and its background. Figure 3.12 (right) shows the results we obtained with our implementation of the local misalignment analysis described in [IAB+96]. This demonstrates clearly that, as pointed out in Section 3.3, differences are more suitable for our purposes than local misalignment analysis.

Figure 3.13 illustrates some results of segmentation, showing selected frames of the foreground sequences. The moving object in "Manuel" is not as sharp as in "Super5", yet the quality of segmentation is still satisfactory.

In order to assess our video coding method, we encoded and decoded the "Super5" and "Manuel" sequences and compared the results with the originals. The left image in Figure 3.14 is a frame of the encoded and decoded "Super5" sequence, whiles the right one visualises the differences between the same frame and the original one.

As a measure of compression quality we used the *point signal to noise ratio* (PSNR) on the difference of each original image of the sequence with the corresponding encoded and decoded one. Given a $n \times m$ source image, f, and a reconstructed image, F, obtained by decoding the encoded version of f,

$$PSNR(f,F) = 20 \ log_{10} \frac{255}{\text{Mean Square Error}(f,F)} \tag{3.1}$$

Figure 3.8: Frames 0, 20, 40 (the last) from the “Super5” sequence.

Figure 3.9: Frames 0, 50, 99 (the last) from the “Manuel” sequence.

where MSE, the mean square error, is

$$MSE = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}(f(i,j) - F(i,j))^{1/2}. \tag{3.2}$$

The plots in Figure 3.15 show that throughout the sequences, the encoding and decoding process has not caused too much image degradation — degradation that increases, as expected, towards the end of the sequence.

3.4.2 Content based manipulation

In this section we give an example of content-based manipulation of a video sequence, in which the segmented representation is exploited to insert a synthetic object into the background, namely an advertising poster. The idea is to edit the background mosaic, then to use the same decoding procedure as described in Section 3.4 to to create a new realistic sequence. The insertion of the synthetic sign is done on the fronto-parallel view

Figure 3.10: Mosaic of “Super5” (background sprite).

Figure 3.11: Mosaic of “Manuel” (background sprite).

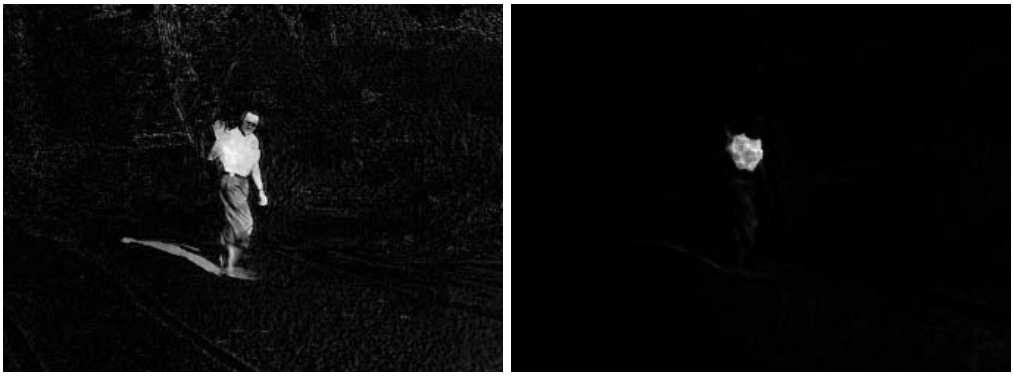


Figure 3.12: Residual analysis for one frame of the sequence “Manuel” with differences (left) and local misalignment (right).

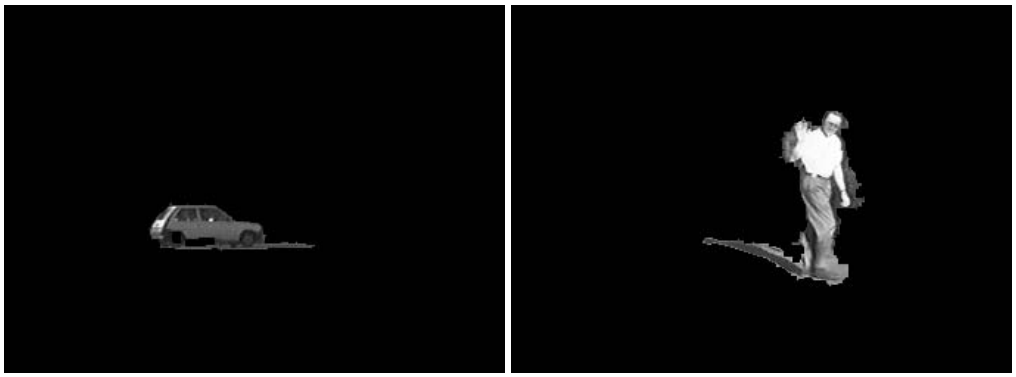


Figure 3.13: Moving objects extracted from the sequences.

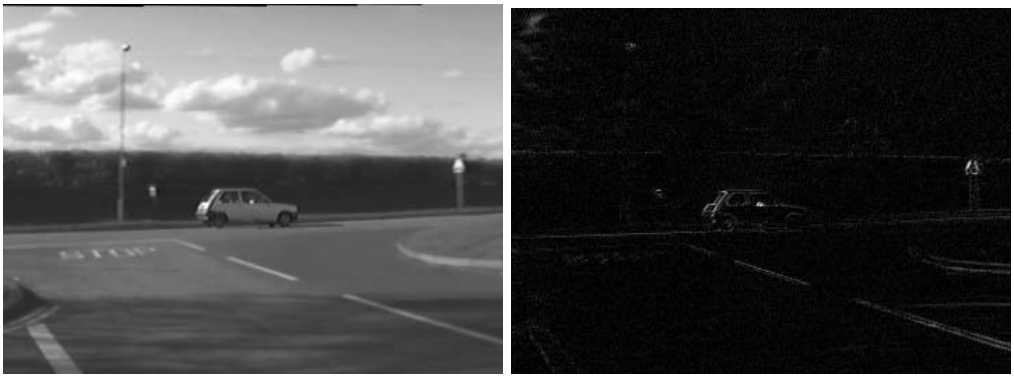


Figure 3.14: Example of a frame from the encoded and decoded “Super5” (left) and differences from the original one (right).

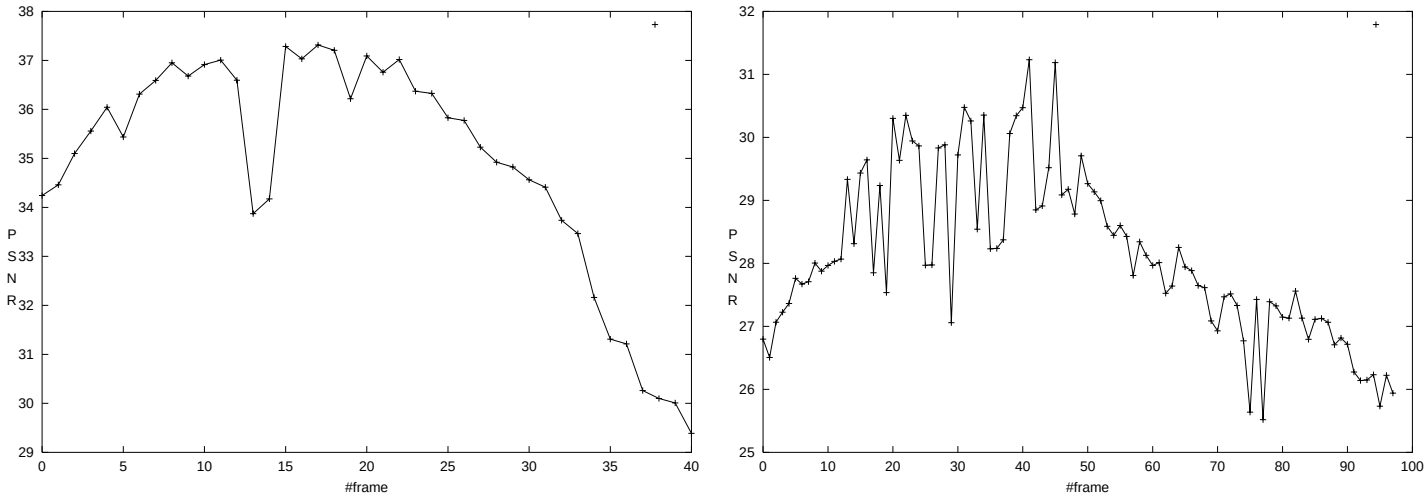


Figure 3.15: Power signal to noise ratio — see text — of the sequence “Super5” (left) and “Manuel” (right)

.

Figure 3.16: Metrically rectified mosaic and a sample frame of the synthetic advertisement sequence.

of the scene (through a *metric rectification* of the mosaic, as described in Section 1.5). After editing, the rectified mosaic is then is warped back onto its original plane, and hence the synthetic sign is slanted accordingly. Figure 3.16 presents a result of such video editing. On the left is the metrically rectified mosaic of the background — all of the editing operations can be performed on this image. Subsequently, a back transformation brings the modified background in its original reference frame, so that the altered sequence can be built correctly (Figure 3.16, right).

Discussion of Part I

Part I was devoted to image sequence analysis issues; we began by reviewing background notions about the geometry of a projective camera, continued by considering the basic aspects of a well known sparse 2D motion estimation technique, and we finally focused on the problem of image registration. Image registration was used to address sequence stabilization (in the very special case that all the camera motion is due to noise) and construction of still mosaics, with and without the presence of moving objects.

This allowed us approach the problem of enhancing the quality of image sequences (Section 2.4.3), representing them in a compact way (Section 3.2), and performing foreground-background segmentation (Section 3.3), which is useful both for video coding (Section 3.4) and for object recognition (Section 3.3).

As will be seen in the Section Conclusions and Future Work, we are currently extending our estimation of the camera motion to the 3D case, in order to classify sub-sequences according to the principal motions the camera underwent during acquisition. This will hopefully help us to discover how to apply object recognition not only to a novel image, but also to a novel image sequence.

Part II

Similarity Measures Based on Hausdorff Distance

Outline of Part II

The first step of object identification consists in representing the object of interest. Usually a model or a set of models is selected, trying to include all the information which could be useful for the recognition task. The recognition is performed by evaluating the similarity between the model, or a part of it, and the test data. In the case of image-based object identification, both model and test data are images, therefore the core problem is the evaluation of similarity between images.

The main contribution of this part of the thesis is a novel similarity method for grey-level images inspired by the Hausdorff distance, that has been derived as a variation of the general correlation-based approach. Its key aspect is to keep the simplicity of correlation methods, while allowing for some degree of geometric deformation and variation on the grey-levels, without assuming any particular transformation model (e.g. locally affine). The resulting method has proved tolerant to occlusions, to small illumination and scene changes, and it is suitable for many applications, ranging from feature tracking, to object identification, and to iconic search.

Chapter 4 introduces the directed Hausdorff distance and Hausdorff distance as measures for point sets, discusses an extension of the Hausdorff distances as functions of translation, and finally gives a brief description of how the Hausdorff distance has been used in Computer Vision.

Chapter 5, after an overview of correlation methods, describes the similarity measure for grey-level images that we propose, and discusses its major aspects and its potential, and

the extension to the multiscale case.

Chapter 4

Comparing binary images

The Hausdorff distance computes the distance between sets of points, and provides a useful measure for matching pairs of edge, texture, and intensity patterns, if they are seen as two sets of points with some special properties and a structure. The versatility of the Hausdorff distance is suggested by the diverse applications in which it appears. In computer vision and image analysis it has been used almost exclusively to match binary pattern. We argue that its applicability can be pushed further, not only to range images [Ols97], but also to grey-level images. In this chapter we give a basic introduction to the directed Hausdorff distance and to the Hausdorff distance (which has been introduced since the former is not symmetric and therefore is not a distance in the mathematical sense). Then we describe how these measure have been extended to deal with 2D points lying on a discrete grid (image points) and used as a similarity measure for binary images in various computer vision applications. For more details on the Hausdorff distance and its application to binary images see [HR93] and references within.

4.1 The Hausdorff distance

In this section we recall the main mathematical concepts about the Hausdorff distance and discuss some of its properties, which will be useful to understand the rest of the chapter. We restrict our attention to finite subsets of $\mathbb{R}^N$.

In a space where the elements are sets, a concept of distance between pairs of elements is

needed. Felix Hausdorff devised a metric function between subsets of a metric space.

Definition 4.1.1 Given two finite point sets A and B of $\mathbb{R}^N$, the *directed Hausdorff distance,* $h(A, B)$, is defined as follows:

$$h(A, B) = \max_{a \in A} \min_{b \in B} ||a - b||. \tag{4.1}$$

where $|| \cdot ||$ is some norm, in our case the L_2 or Euclidean norm.

In practice, the directed Hausdorff distance can be measured in two steps:

1. For a fixed point a of A, compute the distance of a from each point b of B, and select the distance between a and the closest point of B, δ_a (see Figure 4.1, left).

2. Take the maximum of δ_a for all a of A, $h(A, B)$ (see Figure 4.1, center).

From the definition 4.1 we can draw the following observations:

- It *measures the degree of mismatch of A from B*.

- If $h(A, B) = \delta$, then each point of A is within a distance δ from some point of B, and there is one point of A which is exactly at a distance δ from at least one point of B.

- *The directed Hausdorff distance,* is not symmetric and thus it is not a distance. Indeed, in general $h(A, B) \neq h(B, A)$. In the case of Figure 4.1, for example, $h(B, A) < h(A, B)$.

- The directed Hausdorff distance between A and B is zero, if and only if A is a subset of B.

To obtain a distance in the mathematical sense, symmetry can be restored by taking the maximum between $h(A, B)$ and $h(B, A)$. This brings to the definition of *Hausdorff distance.*

Definition 4.1.2 Given two finite point sets A and B of $\mathbb{R}^N$, the *Hausdorff distance,* $H(A, B)$, is defined as follows:

$$H(A, B) = \max\{h(A, B), h(B, A)\}. \tag{4.2}$$

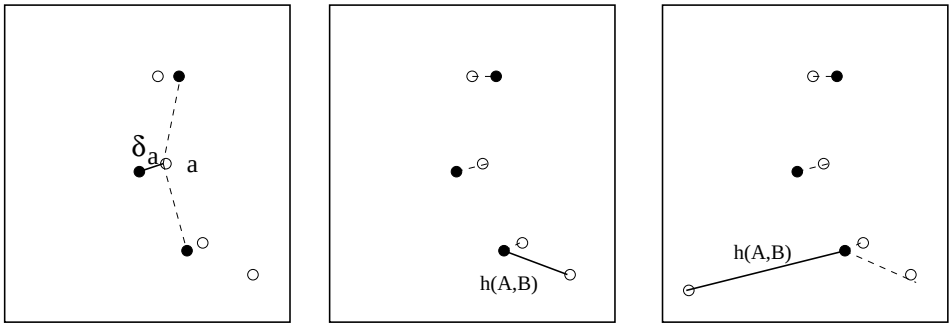


Figure 4.1: Directed Hausdorff distance. Let the empty and filled dots be the elements of the sets A and B respectively. The distance of a point a of A from the set B is denoted by δ_a (left); the maximum of δ_a over all points of A is the directed Hausdorff distance and is denoted by $h(A, B)$ (center). The presence of a single outlier in A is sufficient to distort $h(A, B)$ significantly (right).

For our purposes, a useful property of both distances is that their computation does not involve determining an explicit correspondence between points of A and points of B (many points of A could be close to the same point of B), while an undesirable property is their sensitivity to outliers. Figure 4.1 (right) depicts and example of this problem: one distant element (in the figure the empty dot in the low left) is sufficient to increase the value of $h(A, B)$ (and consequently of $H(A, B)$). The next section shows how this can be countered effectively.

4.1.1 Geometric interpretation

One way to gain intuition on Hausdorff distances is to define them in terms of set inclusion, as described in Figure 4.2 (left). This definition will be very useful on the remainder of the chapter, since the similarity method for grey level images that we are about to describe is strongly inspired by it.

Let B_ρ be the set obtained by replacing each point of B with a disk of radius ρ, and taking the union of all of these disks. Effectively, B_ρ is obtained by dilating B by ρ.

Proposition 4.1.1 *The directed Hausdorff distance $h(A, B)$ is not greater than ρ if and only if $A \subseteq B_\rho$.*

This interpretation suggests an interesting extension of the directed Hausdorff distance, useful to counter the effect of outliers: we discuss the *partial* directed Hausdorff distances [HKR93] by admitting partial inclusions of A inside the set B_ρ, as shown in Figure 4.2. In this case, for example, the presence of the element of A on the low left side of the image increases both $h(A, B)$ and $H(A, B)$. In many applications though, this element would not be enough to conclude that the two sets are different. One possible solution is to compute the greatest subset of A which is included in B_ρ and decide whether this subset is "big" enough for A and B to be "similar". This is especially important in many computer vision and pattern recognition applications in the cases when the scene contains instances of a model which are only partly visible, due to occlusions or failure in the sensing devices, or local illumination changes. With the partial directed Hausdorff distance, instead of taking the maximum $a \in A$ as in Equation (4.1), we rank all the distances between each point a of A and the set B and choose some suitable ranked point. This brings to the following definition:

Definition 4.1.3 Let q be the cardinality of set A. The k-th partial directed distance, $1 \leq k \leq q$, is given by the k-th ranked point in the set of distances:

$$h_k(A, B) = k_{a \in A} \min_{b \in B} ||a - b||. \tag{4.3}$$

For example, the q-th ranked value is the maximum, and the $q/2$-th is the median.
$h_k(A, B) = \rho$ tells us that k points in A are within a distance ρ of some point of B. Let A_k be a subset of k points of A, and ρ an acceptable value for the Hausdorff distance between two sets. Let m be the greatest value so that some A_m (a subset of A of cardinality m) is contained in B_ρ. While the directed Hausdorff distance $h(A, B)$ could be increased by the points of A which do not belong to A_m, $h(A_m, B) = h_m(A, B)$ is still not greater then ρ.
If m is sufficiently large, the elements of A at a distance from B greater than ρ can be seen as outliers. By choosing an adequate lower bound for m, potential outliers can be ignored. In order to compute the partial directed Hausdorff distance we normally specify some fraction $0 < f_1 \leq 1$ of the points of A which are to be considered and set $k = \lfloor f_1 q \rfloor$,

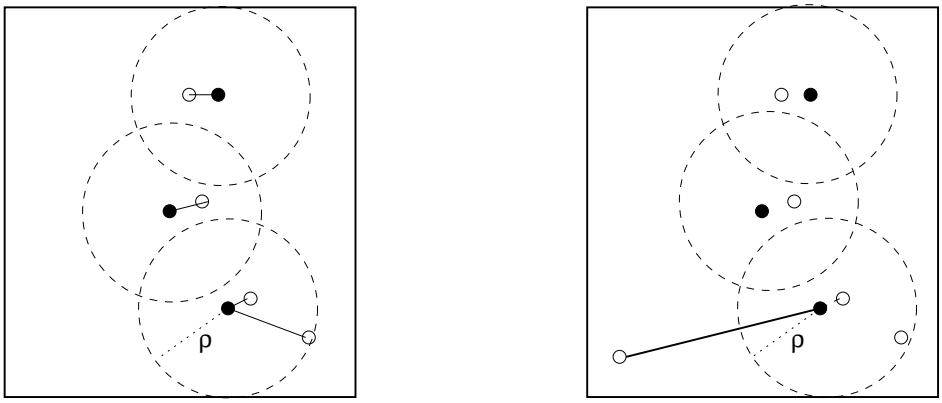


Figure 4.2: Geometric interpretation of the directed Hausdorff distance. Let again the empty and filled dots be the elements of the sets A and B respectively. Let B_ρ be the union of the set of disks of radius ρ centered at the points of B. From the left figure it is clear that $A \subseteq B_\rho$ if $\rho = h(A, B)$ and *vice versa*. The right figure shows an example of how this property, through the concept of partial inclusion, can be used to overcome the sensitivity to outliers.

that is we seek the distance where some given fraction f_1 of the points of A lie near the points of B. As pointed out in [HR93] one key property of h_k is that it does not require to pre-specify which part of set A is to be compared with set B, because the computation of the directed Hausdorff distance determines how far each point of A is from the nearest point of B, and thus automatically selects the k points of A which are closest to B.

Definition 4.1.4 The partial Hausdorff distance is defined as:

$$H_{lk}(A, B) = \max\{h_l(A, B), h_k(B, A)\} = \max\{h(A_l, B), h(A, B_k)\}.$$

This function does not satisfy metric properties, but it does obey weaker intuitive conditions: that metric properties are satisfied between given subsets of A and B (of size l and k respectively). Since our approach is more related to the directed distance than the bidirectional one, we will not give further detail about this property — the interested reader can refer to [HKR93].

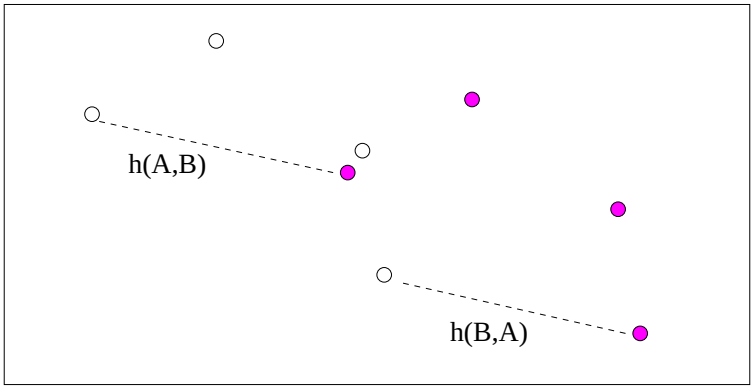


Figure 4.3: An example of how a translation can affect the Hausdorff distance. Set A is represented by the empty dots, set B by the filled ones. Set B is a copy of set A, where each point have been translated of $-\mathbf{t}$. Even if the two sets where identical before the translation, after the translation $h(A, B)$ and $h(B, A)$ are not equal to 0. The fact that both distances are actually equal to $\mathbf{t}$ is only a coincidence of this example.

4.1.2 Hausdorff distance as a function of translations

The Hausdorff distance measures the mismatch between two sets or parts of them at fixed positions with respect to one another, in some common reference frame. In the setting we are about to present, instead, it is much more useful to measure the mismatch between all the possible relative positions of the two sets [1]. We focus primarily on the case where the relative positions of sets are described by a group of translations,

$$S_B(t) = H(A, B \oplus t),$$
$$S_A(t) = H(A \oplus t, B),$$

where $\oplus$ is the vector notation: $X \oplus t = \{x + t | \ x \in X\}$.

We can also define the minimum value of the Hausdorff distance between sets A and B in the following way [HKR93]:

$$G(A, B) = \min_t H(A, B \oplus t) \tag{4.4}$$

[1] If the sets are finite and live in a discrete space — the grid of pixels, for instance — it is not difficult to try all the possible relative positions.

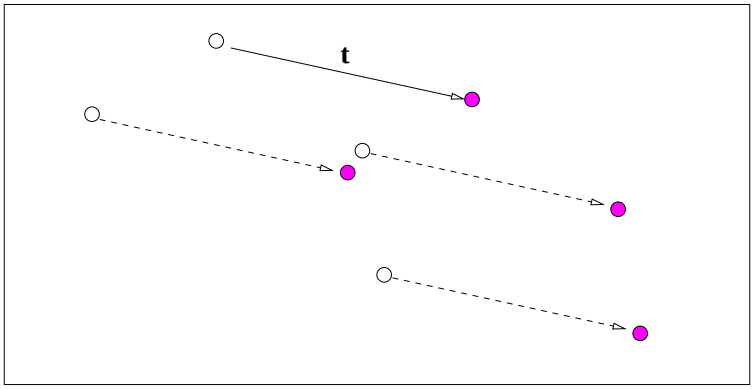


Figure 4.4: If we compute the Hausdorff distance allowing for translations, we find that for a translation equal to $\mathbf{t}$ the minimum value of the Hausdorff distance is 0.

In Figure 4.3 the set A is represented with empty circles and the set B with filled circles. B is just a translated version of A, but their Hausdorff distance is not negligible. Instead, $G(A, B) = 0$ (see Figure 4.4).

4.1.3 How to compute $H(A, B)$ and $G(A, B)$

In this section we consider the special case where the points of A and B lie on a plane, and in particular $A, B \subseteq \mathbb{R}^2$. From Equations (4.1) and (4.2), we derive

$$H(A, B) = \max \left\{ \max_{a \in A} \min_{b \in B} ||a - b||, \max_{b \in B} \min_{a \in A} ||a - b|| \right\}.$$

If we define $d(x) = \min_{b \in B} ||x - b||$ and $d'(x) = \min_{a \in A} ||a - x||$, then

$$H(A, B) = \max \left\{ \max_{a \in A} d(a), \max_{b \in B} d'(b) \right\}.$$

The graph $\{(x, d(x)) | x \in \mathbb{R}^2\}$, is called *Voronoi surface* [HKS91]. This surface gives for each location x the distance from x to the nearest point $b \in B$. For points in the plane it can be visualized with local minima of heigth zero in correspondence of each point $b \in B$, it has the property that its local maxima are equidistant from two or more local minima (for this reason we refer to it as Voronoi surface, by analogy to Voronoi diagrams that specify the locations equidistant from two or more points of a given set [PS85]).

In a similar way, we can re-write the Hausdorff distance as a function of a translation:

$$\begin{aligned} S_B(t) = H(A, B \oplus t) &= \max\left\{ \max_{a\in A} \min_{b\in B} ||a-(b+t)||, \max_{b\in B} \min_{a\in A} ||a-(b+t)|| \right\} \\ &= \max\left\{ \max_{a\in A} \min_{b\in B} ||(a-t)-b||, \max_{b\in B} \min_{a\in A} ||a-(b+t)|| \right\} \\ &= \max\left\{ \max_{a\in A} d(a-t), \max_{b\in B} d'(b+t) \right\} \end{aligned}$$

that is, $H(A, B \oplus t)$ is the maximum of translated copies of the distance transforms $d(x)$ and $d'(x)$.

4.2 The Hausdorff distance in Computer Vision

Hausdorff measures have been used in computer vision nearly exclusively to match binary patterns of contour or edges. It is easy to see how a edge map can be seen like a set of points lying on the Euclidean plane: each edge pixel simply becomes a point of a set lying on a grid. It is less direct to interpret in a similar way grey-level images or range images. In this section, after mentioning some applications of the Hausdorff distance to match edge maps, we will see how it has been used with range images. The next chapter will be dedicated to a method we propose to extend these approaches to the case of grey-level images, thanks to a suitable representation of grey-level images with binary sets of 3D points.

4.2.1 The Distance Transform

In this section we first consider the case of computing the Hausdorff distance on sets of points lying on discrete grids. In this way it could be applied to binary images, such as edge points maps. We start by a change of notation, introducting the *model* M, a binary image of size $m_r \times m_c$, and the *image* I, a binary image of size $d_r \times d_c$, bigger or equal to M. These images contain a 1 at each edge point. We seek an occurrence of M inside I, by comparing $h(M \oplus t, I)$, for each possible translate t:

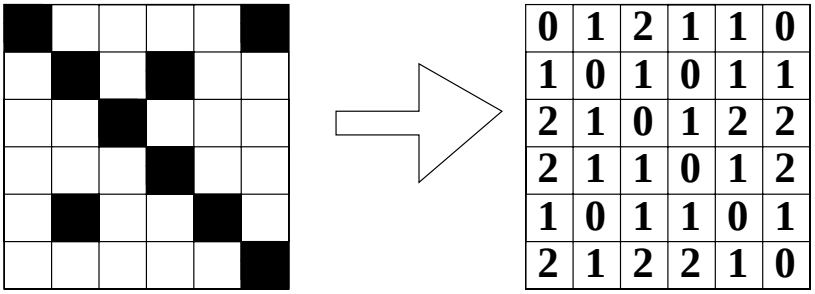


Figure 4.5: On the left an example of a binary map - the edge points are the black pixels - and on the right its Distance Transform, computed with one possible approximation of the Euclidean distance.

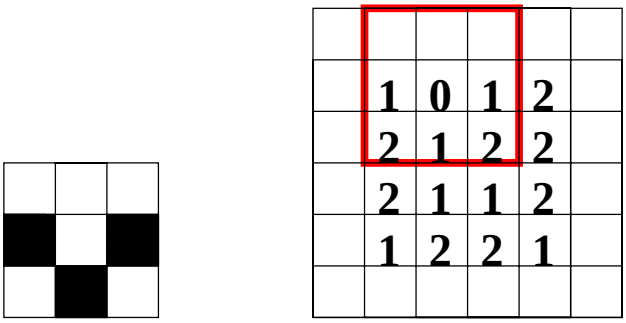


Figure 4.6: A model (on the left) and the corresponding $s(t)$ (on the right).

$$\begin{aligned} s(t) &= h(M \oplus t, I) = \\ &= \max_{m \in M} \min_{i \in I} ||m + t - i|| \\ &= \max_{m \in M} d'(m + t) \end{aligned} \tag{4.5}$$

If the points lie on a grid (as in the case of images), the Voronoi surface $d'(x)$ is also known as *Distance Transform* [Bor84], because it gives the distance from any point x on the grid to the nearest point in the set I.

We can imagine to compute $s(t)$ by placing $M \oplus t$ on the top of the Distance Transform of I, and looking for the largest value of the surface among the ones "covered" by a point of $M \oplus t$. The minimum value of $s(t)$ identifies the best match of M in I. Figure 4.5 shows a simple example of Distance Transform (right) computed with the block distance, of the binary image I on the left. In Figure 4.6 small binary map on the left is the model.

On the right we show the corresponding to the $s(t)$ of Equation (4.5). The thicker square shows the lower value of $s(t)$, that corresponds to the region of I most similar (in this case identical) to M.

4.2.2 The minimum Hausdorff distance for binary maps

The work by Huttenlocher and his co-workers [HKR93, HR93, Ruc97] is the most representative and complete introduction to the Hausdorff distance in computer vision. They used the directed Hausdorff distance to implement an efficient search of a binary edge model M in a binary edge image I, and then studied the problem of allowing different tranformation to the model M, $t(M)$, mainly translation and scale variation. They computed $s(t) = h_k(t(M), I)$ and seeked values of the surface which were above a fixed threshold. They also experimented various ways of computing distance tranforms, and proposed a multiresolution variant of their method and various heuristics to improve the efficiency of the algorithm.

A problem of this method applied to edge images is that, if the edge density in I is high, the likelihood of a successful match against *any* $t(M)$ is also high. To obviate this, the same procedure is applied a second time, only on promising matches, after swapping image and model in a way which recalls of the left-right consistency constraint of stereo matching: for each point which passed the similarity test on the forward procedure they apply a reverse procedure, by computing $h_k(M, t(I))$; the authors describe this approach as if a sort of hypothesize and test method. In the next chapter, once we have introduced our method for estimating similarity between grey-level images, we discuss the connections between our approach and the one proposed by Huttenlocher, and explain the differences between working with edge maps and grey-level images.

4.2.3 Applications

The versatility of the Hausdorff distance at evaluating similarity between point sets is suggested by the variety of applications in which it appears: defect detection [DY99], gesture recognition [KSPF96], robot localization [SD99], range image analysis [Ols97], and

content-based video and database indexing, object tracking [NNT01].

In [HR93] it has been applied to the object detection on edge maps computed from indoor images, and on binary scans of engineering drawings. In [DY99] it is used as a method for defect detection.

In [KSPF96] the objective is to recognize motion and gestures of humans; specifically the task is to understand if the person is pointing at something. The Hausdorff distance, applied to edge maps, is used to localize the presence of a person within a scene, and to find the position and the orientation of the head and the hands; and finally, after the area the person is pointing at has been localized, it is used to identify the object.

Olson presented an algorithm which used the Hausdorff distance to match three-dimensional surface maps [Ols97]. To our knowledge, it is the only work which attempts to use Hausdorff distance to match more complicated data than binary maps. In [Ols98] he proposes a probabilistic formulation of image matching which generalizes a version of Hausdorff matching, in order to model feature uncertainties and prior knowledge. In both works the application is robot self-localization.

Chapter 5

Comparing grey level images

In this chapter we introduce the problem of finding similarities between grey-level images, bearing in mind our target problem of object detection. In particular, we study a correlation method which we derive from the Hausdorff distance. The attractiveness of the method lies in its tolerance to small geometric deformations and illumination changes, and also to local object variations and occlusions. We discuss variants and extensions of the method, which take into account efficiency and quality issues, analyse the relationship with the similarity measures for binary images described in Chapter 4, and conclude with experimental results and with an evaluation of the influence of the parameters of the method on the results obtained.

5.1 Introduction

In problems like image-based object recognition and identification, an object can be represented by a selection of images which describe various object details or poses. These images are often called image models or simply models, as they describe the object in question. They are a visual description of the object and can be used to find occurrences either of the same object in another image, or of a similar object in another image. The more complete the description is, the more likely the identification will be correct. In view-based systems a good description should contain all the representative views and all the characterizing details. In this chapter we assume that we have a model for the

description of an object and wish to find an occurrence of it in a specific image.

The existing literature on similarity measures for images is extensive — even trying to classify the existing methods is not an easy task. One possible classification divides the techniques into feature-based ones and area-based ones: although this classification is often incomplete and unsatisfactory, since there are methods which fall in between the two classes or outside their domain, it is still the one commonly used.

Feature-based methods require a preliminary representation of the model in terms of some features or some descriptors, and consist of finding similar features in the images analyzed. In case of face detection, for instance, a face can be described by the contours of eyes, mouth, and nose, and faces can be detected by looking for similar shapes inside images.

The *area-based approach* consists of a comparison of grey-level patterns, without any preliminary interpretation. In some instances of this approach, one could say that area-based methods are just another example of feature-based methods, the features being areas of the image. When the similarity criterion chosen to compare areas of two different images is correlation based, we refer to the so called *correlation* methods. Since the area-based similarity measure presented later in this chapter can be classified as a correlation technique, we begin with a brief introduction to correlation methods.

The similarity method we propose was originally inspired by the concept of Hausdorff distance, and has been derived as a variation of the general correlation approach. The key idea is to retain the simplicity of correlation methods, while allowing for a certain degree of deformation and light variation. The correlation is still performed on a rectangular shaped window, but pixel by pixel comparison allows the method to accept partial matches, and to tolerate occlusions and discontinuities. Thanks to its versatility, we have used it successfully for stereo matching and feature tracking [OTV01b, OTV01a], and image search. One of its interesting features is that no explicit numerical computations are necessary, so that the method is also suitable for high-speed implementations. It extends naturally to multiscale, thus improving in efficiency.

5.2 Correlation methods revisited

Many classical problems in computer vision, such as sequence analysis, stereo correspondence, feature matching, object recognition, can be regarded as instances of a correspondence problem. Methods based on correlation measures represent a vast class of possible approaches to solving correspondence problems, but, unlike more sophisticated techniques, which may take into account possible transformations of the scene [PZ98, ST94], look at invariant representations [BS00, Li92, SM97], or rely on prior image transformations which exploit local ordering information [BN98, ZW94], direct correlation methods simply match pairs of elements, using similarity measures to describe the degree of correlation between them. Since the elementary item of an image, the pixel, does not carry enough information to make it distinguishable among the others, most methods use aggregates of pixels to perform their similarity tests. For efficiency reasons the most popular aggregates are windows of a fixed rectangular shape, which unfortunately do not behave nicely in presence of occlusions or object borders, as pointed out in [BVZ98].

A general definition of correlation methods is the following: given a rectangular window M of an image I, the corresponding rectangular window M' of an image I' is found by computing the maximum of

$$\Psi(\mathbf{t}) = \mathcal{F}(M, M'_{\mathbf{t}}) \tag{5.1}$$

with respect to $\mathbf{t}$, with $\mathbf{t}$ a possible translate of M' from the position of M, and $\mathcal{F}$ some chosen *score* function, $\mathcal{F} : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$. The search region can be the whole image (if there is no a priori knowledge) or a subset of it (for instance an epipolar line for stereo). Given two windows W_1 and W_2, instances of M and $M'_{\mathbf{t}}$, two well-known examples of $\mathcal{F}$ are

$$\mathcal{F}(W_1, W_2) = \sum_p W_1[p] W_2[p], \tag{5.2}$$

(the pixel p ranges over the image window), which yields the *cross-correlation* between W_1 and W_2, and

$$\mathcal{F}(W_1, W_2) = -\sum_p (W_1[p] - W_2[p])^2, \tag{5.3}$$

which performs the *Sum of Squared Differences* (SSD) or *block matching*. In many practical situations SSD is preferable since, unlike cross-correlation, it is not biased by the presence of regions with very high or very low intensity values [TV98]. For this reason cross-correlation is often replaced by *normalized cross-correlation*: if N_1 and N_2 are the sizes of W_1 and W_2 respectively, then

$$\mathcal{F}(W_1, W_2) = \frac{\sum_p (W_1[p] - \overline{W}_1)(W_2[p] - \overline{W}_2)}{N_1 N_2}, \tag{5.4}$$

where

$$\overline{W}_1 = \frac{1}{N_1} \sum_p W_1[p],$$

and

$$\overline{W}_2 = \frac{1}{N_2} \sum_p W_2[p].$$

Both SSD and cross-correlations, given an image window M, look for the closest translate $M'_{\mathfrak{t}}$, assuming that no transformation except translation can occur between the two images. To take into account grey level changes within a fixed interval, or small local transformations of the image window (for instance small scale variations or small affine transformations), a simple possibility is to allow for some variations of the target image I', both in the spatial positions of the pixels and in their grey levels. The method described in the next section follows this approach. It is based on the following function:

$$\mathcal{F}(W_1, W_2) = K_H(W_1, W_2) = \sum_p \theta(\epsilon - \min_{q \in N_p} |W_1[p] - W_2[q]|) \tag{5.5}$$

where θ is the unit step function. This score function counts the number of pixels p in W_1 which are within a distance ϵ (in the grey levels) from at least a pixel q of W_2, with q in a neighbourhood N_p of p. Notice that, in general, this function is not symmetric, unless N_p coincides with p. This idea may recall the definition of *shuffle tranforms* [Kut00], so the relationship between the two will be addressed later in the chapter, after the description of our method.

To apply correlation measures, the content of each window should ideally be smooth

(i.e., it should not overlap boundaries), but of course this is not always the case. Methods based on the comparison of rectangular shaped windows often lack precision at object boundaries, and do not generally tolerate well the presence of occlusions. This problem has been widely addressed already, and many solutions have been proposed, which range from using robust statistics [BBW88, MMRK91], to adaptive windows [KO94, FRT97], or statistical methods to select connected windows of arbitrary shapes [BVZ98].
In our approach the correlation is still performed on a fixed rectangular shaped window, but we use a pixel by pixel comparison and produce a correlation value by counting all the pixel which passed the test (see Equation (5.5)). This allows us to spot occlusions, accept partial matches, and tolerate occlusions and discontinuities. The next section provides further details.

5.3 The proposed similarity method

The core of this problem can be described in the following way: given two regions W_1 and W_2 we wish to decide whether they are similar or not; we also wish to be able to fix the degree of similarity required, by choosing, for instance, an interval around the grey levels of W_1 within which the grey level values of the window W_2 can vary.
If we consider each pixel of the two windows and their grey values as points in some space and the two windows as sets of points, then we could compute the Hausdorff distance between these sets and then express the similarity between the windows as a function of this distance. Computing Euclidean distances is known to be a computationally expensive task, and numerous measures which are easier to compute have been proposed in the literature.
Our method introduces a drastic but rather effective binary classification of distance: for each set point we simply decide whether it is close to some point of the other set or distant. Apart from this simplistic version of a distance transform, our approach follows faithfully the algorithm described in Section 4.2.1.

5.3.1 Approximating distances with a binary measure

Starting from two point sets, instead of computing the Hausdorff distance between the two of them, an efficient alternative approach is to fix a maximum distance (or, equivalently, a minimum degree of similarity) accepted between two sets, and then to see if the two sets in question are closer than this distance or not. When comparing two image windows, this approach is expressed by Equation (5.5). In this case the maximum distance is fixed by ϵ and N_p: we can allow for variations in the grey level of p (within ϵ) or in its localization (inside N_p). A practical way of implementing Equation (5.5) is to reason in terms of dilation and inclusions instead of inequalities (analogously to the step from the formal definition of directed Hausdorff distance to the more intuitive definition based on set inclusions). Note that Equation (5.5) was inspired by the directed Hausdorff distance and therefore it is not symmetric. Again, to restore symmetry:

$$\mathcal{K}_H(W_1, W_2) = \max\{K_H(W_1, W_2), K_H(W_2, W_1)\} \tag{5.6}$$

Following this approach, K_H can be computed in four steps.

1. Expand W_1 into a 3D binary matrix $\mathcal{W}_1$, the third dimension being the grey value. That is, for i and j spanning the pixel locations and g the grey values:

$$\mathcal{W}_1(i, j, g) = \begin{cases} 1 & \text{if } W_1(i, j) = g; \\ 0 & \text{otherwise.} \end{cases} \tag{5.7}$$

   In order to limit the space required, the number of grey levels G of the 3D expansion is equal to the number of grey levels actually present in the model.

2. Build the 3D binary matrix $\mathcal{W}_2$ from W_2 in the same way, using the same G for the size of the third dimension.

3. Dilate the matrix $\mathcal{W}_2$ by growing its nonzero entries by fixed amounts ϵ_r,ϵ_c in the space dimensions[1] and ϵ in the grey value dimension. Let $\mathcal{D}$ be the resulting 3D

[1] Note that N_p of Equation (5.5) is a $((2\epsilon_r + 1) \times (2\epsilon_c + 1))$ neighbourhood of p.

binary matrix. The dilation varies according to the degree of similarity required and the transformations allowed, and can be different in all the three directions.

4. Compute the size of the intersection between $\mathcal{W}_1$ and $\mathcal{D}$, and call it s.

This value s represents the number of pixels of W_1 which are within a fixed distance (given by the choice of ϵ and N_p) from W_2, reminding us of the definition of partial directed Hausdorff distance.

The dilation along the grey values allows for small illumination changes, and takes into account the acquisition noise. In the same way, the dilation along the spatial dimensions deals with small scale changes and small deformations such as those produced by camera motion.

5.3.2 Finding occurrences of a model in an image

Once the core issue of comparing two regions belonging to two different images has been formalized, the problem of finding occurences of a model in an image can be stated as follows: Let M be a $m_r \times m_c$ model image. We wish to find its best match with a $m_r \times m_c$ window of I (the size of which is $dim_r \times dim_c$).

The algorithm described in the previous section can be extended to measure the similarity between all the possible relative positions of two windows, in an analogous way to the one described in Section 4.1 for the general continuous case. This is equivalent to taking the smaller image (usually the model) and moving it about the larger image, computing $K_H(M_{\mathbf{t}}, I)$ for all possible translates $\mathbf{t}$ (equivalent to what is discussed in Section 4.2.1).

This extension can be realized in two different ways: The first one is to crop a window of the same size as M based at position $I(\mathbf{t})$, and then to apply the algorithm to M and the subregion of I exacly as described before. The second way is to apply the dilation to the whole of I first, and then to scan with $\mathcal{M}$ the space occupied by $\mathcal{D}$ for all the possible translates $\mathbf{t}$.

The advantage of this second approach is twofold: firstly, the processing time is reduced, by combining all of the dilation in a single step at the beginning — where it effectively becomes preprocessing; secondly, this choice exploits the contiguity properties of the image,

since the grey-level patterns of an image region tend to be influenced by the neighbouring regions (consider shadows, for instance). For the second way the algorithm is extended in the following fashion:

1. Expand M into a 3D binary matrix $\mathcal{M}$, as described in Equation (5.7).
2. Similarly expand the whole image I obtaining the 3D binary matrix $\mathcal{I}$.
3. Dilate $\mathcal{I}$ as described earlier obtaining $\mathcal{D}$.
4. For each possible translate $\mathcal{M}_{\mathbf{t}}$, compute the size of its intersection with $\mathcal{D}$: this produces a discrete surface $s(\mathbf{t})$.

If this surface is normalized with respect to the area of the model, each element gives the *match value* between the corresponding region and the model. The match value, which belongs to the closed interval $[0, 1]$, is equal to 1 if the model is equivalent to a region of the image, up to a difference allowed by the dilation.
Once a candidate match is found, a further check on the connectivity of the aggregate of pixels which passed the test can be performed to distinguish between occlusions and bad matches.
From the point of view of complexity evaluation, it can be shown that with an appropriate choice of the data structures, the algorithm is reduced to a set of entry-wise logical AND operations between $\mathcal{M}$ and $\mathcal{I}$ and no numerical computations are required. From an implementation point of view, it may be worth pointing out that an explicit computation of both $\mathcal{M}$ and $\mathcal{I}$ is not necessary, but for each possible translate M itself can be used to access the appropriate entries of $\mathcal{I}$. This reduces the time complexity from $O(m_r \times m_c \times G \times (dim_r \times dim_c))$ to $O(m_r \times m_c \times (dim_r \times dim_c))$.

Relationship between the algorithm and the Hausdorff distance

There are a few observations that can be made about the method in relation to the directed Hausdorff distance.
If (a) the dilation of the matrix $\mathcal{I}$ is isotropic in an appropriate metric, and (b) $s(\hat{\mathbf{t}})$ takes on the maximum possible value — that is, $\mathcal{M}_{\hat{t}} \subseteq \mathcal{I}$ — then the directed Hausdorff distance

between $\mathcal{M}_{\hat{\mathbf{t}}}$ and $\mathcal{I}$, $h(\mathcal{M}_{\hat{\mathbf{t}}}, \mathcal{I})$, is not greater than ρ. In general, if $s(\hat{\mathbf{t}}) = k$, then the k-partial directed Hausdorff distance between the same sets is ρ. Loosely speaking, we choose an acceptable distance between two sets, and then find whether the sets we are comparing, or subsets of them, are within that distance.

Once the grey level images are expressed as a 3D binary structure, the relationship with Huttenlocher's method should be clear. Unlike the 2D binary image case, though, the reverse operation to eliminate spurious matches is not necessary, thanks to the intrinsic properties of the set of points (see Section 4.2.2). In the 2D binary case, if the image is a rich distribution of edges, this is equivalent to having a dense set of points which would match with every possible model. In 3D this would be equivalent to a 3D image structure full of 1s, which is impossible by construction, since the 3D structures we have defined are effectively 2D$\frac{1}{2}$ grids, and, for each vertical direction, only one entry is set to 1 before dilation.

Another reason for the success of the method is the autocorrelation property of most parts of grey level surfaces. In fact, with the exception of object boundaries, grey levels tend to vary smoothly across images, making more unlikely the presence of spurious matches.

Relationship with other correlation methods

With respect to other correlation methods, a few remarks are in order. Instead of thinking in terms of the average distance between windows' grey levels, we perform pixel-wise similarity tests. In presence of a severe dilation along both the grey level dimension and the spatial ones, this method can lack precision in the localization, but it is less likely to produce false matches in case of occlusions or multiple matches. Also, given the best match, it is straightforward to determine which pixels passed the similarity test, and refine the matching or improve the localization. In this sense it could be seen as a simple method to compute variable size aggregates of coherent pixels. Unlike [BVZ98] which produces variable size windows that cut off occlusions by applying an hypothesis test to each pixel of an inizial window, our method is more a qualitative tool to spot object boundaries and occlusions, than a general recipe to overcome the problem. We still use

fixed size rectangular windows and allow for partial matches (i.e., matches of subsets of the windows) if it is necessary.

Finally, notice that the computation of $h(A, B)$ does not involve determining an explicit correspondence between points of A and points of B (many points of A could be close to the same point of B). This contrasts with most correlation functions and also with most model-based recognition methods which do determine a correspondence between points of the model and points of the image.

Comparison with the shuffle transforms

The idea of matching two image regions by fixing tolerances in space and grey-levels can be found also in [Kut00], with the definition of *shuffle transforms*. Shuffle tranforms are used for an implicit definition of "approximate" 3D shapes, in case no consistent 3D shapes can be reconstructed (for instance, in N view stereo, in the presence of calibration errors).

A 2D transformation $\mathcal{T} : I_1 \rightarrow I_2$ is called a *r-shuffle*, if for each point of the image I_2 we can find a point of "identical color" (within a fixed threshold) within a disk of radius r in I_1.

A volume is defined r-consistent to a set of N views, if for each view I_i there exists a r-shuffle I_i' which is a full projection of the 3D shape (the so called *photo consistency* [KS99]).

In our case, if after a dilation r in the space directions (and a dilation on the grey-levels which is not specified) we find a correspondence between each point of image I_1 and *each* point of image I_2, then we can say that I_2 is an r-shuffle of I_1. However, our similarity function does not necessarily seek correspondences with each point of I_2 (since, for example, all points of I_1 could match the same point of I_2), so it is not always true that if $K_H(I_1, I_2)$ with a dilation r is equal to the maximum value then I_2 is an r-shuffle of I_1.

Another difference between the two approaches seems to be related to the application domain: in our case both I_1 and I_2 are existing images and we wish to verify whether their difference is within a threshold which we find acceptable. Instead, in the the case of shuffle transform, I_1 is an existing view, r is a threshold for all the possible transfor-

mations allowed, and the shuffled images are some synthetic transformations of I_1 which represent implicitly possible tolerances on the final 3D shape.

5.4 Evaluation of the method

The method described is a variation on correlation methods and has interesting properties: it is tolerant to occlusions (since it accepts partial matches) and it can also identify the occlusions, it tolerates small scene deformations or illumination changes and thus, in the case of small deformations, it does not require us to parameterize or to estimate them. The method, as it stands, is a good compromise between simplicity and effectiveness, but there are a few aspects which could be improved, and these will be discussed in the reminder of this section.

Figure 5.2 shows[2] the results of an experimental evaluation on a set of images acquired in our laboratory. During acquisition, we have deliberately introduced some variations in the appearance and the illumination of the scene, and in the camera position, in order to understand the "reactions" of the method. The model image (Figure 5.1) includes an object (the mask) which was removed from the scene before capturing the test images (Figure 5.2), so to create an occlusion. In Figure 5.2, from top to bottom, we show four possible changes: the image on the top has been acquired from a similar position to that of the model, and all the changes are due to natural causes except from the object removed. The second image was acquired after a small rotation of the camera about the optical centre. Before the acquisition of the third image the room illumination was changed (switching on a desk light by the side of the scene). Finally, the fourth image was acquired after a small translation of the camera. The results can be assessed by analysing the *positive scores distributions*[3]; in particular the distribution of the bottom example shows thick lines of unmatched points along the objects boundaries, which are the effects of parallax due to the camera translation.

[2] In the following examples a rectangular shape is drawn around each best match.

[3] The $m_r \times m_c$ maps that will be called *positive scores distributions* represent a 2D projection of the intersection between $\mathcal{M}$ and $\mathcal{D}$ (given a possible matching window, the white pixels of the map correspond to the pixels of the window which passed the test).

Figure 5.1: Model used for the experiments shown in Figure 5.2

This procedure is too time consuming if the search operation has to be repeated many times, as in content based image retrieval, where the set of the model can be a big image database.

Another issue is that the scale of the model is fixed, but can vary with the instances of the object inside the test images. We must deal with the problem that the test image will not necessarily represent the object at the same scale as the model. Note, however, that our method tolerates small scale variations, as they can be conflated with spatial misalignments: Figure 5.3 shows the effects of searching for a model (of a window) across a zooming sequence; the results are correct even though the images contain similar patterns (other images).

The illumination issue is a major problem for almost all similarity method based on images, both feature based (since the illumination changes can affect the extraction of features) and area-based. Our method, as it stands, can cope with a certain amount of variation in illumination, thanks to the dilation of the grey levels. If the illumination change is partial, this can also be tackled thanks to the tolerance to occlusions (since this variation can be seen as a local occlusion). Nevertheless, the robustness to light variations of the method can be improved without affecting its time complexity, as described later in this section.

5.4.1 A Multiscale approach

By multiscale approach we actually mean a combination of preprocessing operations and data structures which help to deal with computation optimization and scale variations. The two things are mixed together, even though they represent distinct phenomenona,

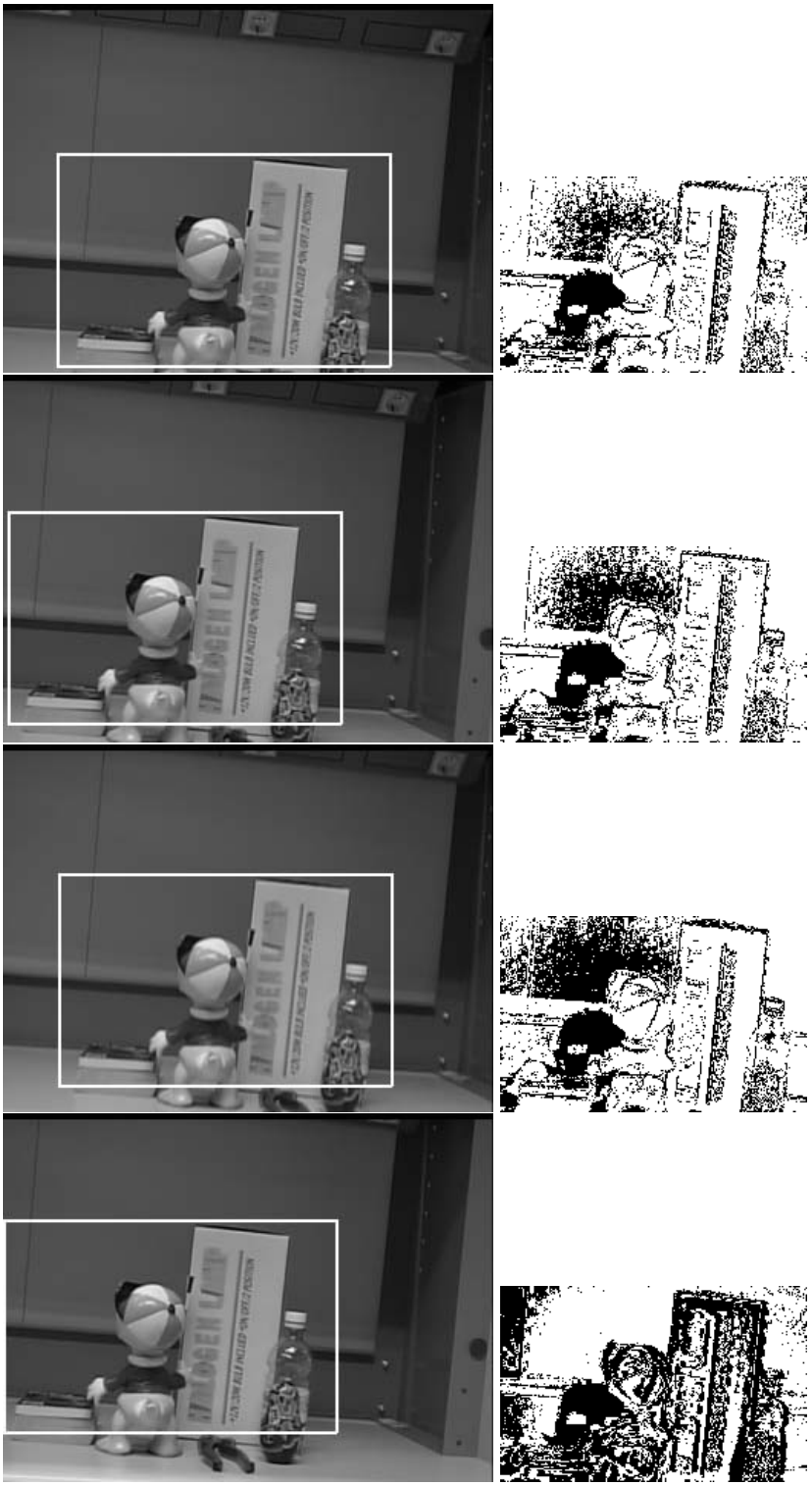


Figure 5.2: Test images and the corresponding positive scores distributions which describe the results of matching against the model in Figure 5.1 (notice that one object — the mask — has been removed). From top to bottom: variations produced by time, by the rotation of the camera, by an abrupt light variation, by a translation of the camera.

Figure 5.3: Search for a detail (a window) at different scales, with the algorithm of Section 5.1. The quality of the localization decreases but the results are still acceptable, since the spatial dilation takes care of the negative effect caused by a limited scale variation.

only for efficiency reasons.

We deal with possible scale changes between the models and the test images by replicating each model at various scales, as shown in Figure 5.4: with a warping operation the original model is reproduced at different scales and all these new images are used as combined models.

This approach is an alternative to the ones which apply scale transformations to each of the image regions examined. Our approach has a lower time complexity but a higher space requirements, and it has been chosen only because in the applications in which we have used it, the time issue was more important than the space issue.

Whenever the same view is replicated in different models (at different scales), multiple matches can be found. In case more than one match is detected in a neighbourhood, the one with the higher match value is kept and the others are discarded.

The use of more models at different scales increases the computation time. To deal with such time complexity issues, each image (both models and test images) is represented by a Gaussian pyramid like the one shown in Figure 5.5. The search is first performed at the coarser level and it is refined at a higher level only if the resulting match value is above a certain threshold. This approach is based on the fact that the match value between two fine images cannot decrease at coarser levels. The use of the Gaussian pyramid representation causes a significant reduction of the computational cost: indeed, assuming that the size of the model is $n \times n$, and that the size of the image is $(n+m) \times (n+m)$, then a simple search would require $(m+1)^2 n^2$ operations. Instead, with a Gaussian pyramid representation the number of operation decreases to $[(\frac{1}{2^r}m + 1)^2 n^2]/4^r$, where r is the number of the lowest level reached in the Gaussian pyramid representation. Note that, for each level added the time complexity reduces by a factor 4.

5.5 Sample applications and results

In this section we exhibit some applications of this method to stereo and motion correspondence and object search. In the first two applications the method is used as a local correlation measure, while in object search and identification it is used as a global sim-

Figure 5.4: Original model (left) replicated at different scales.

Figure 5.5: Gaussian pyramid representation

Figure 5.6: Representation of the amount of information contained at different levels of the Gaussian pyramid

ilarity measure. We recall that the *positive scores distributions* are $m_r \times m_c$ binary maps that represent a 2D projection of the intersection between $\mathcal{M}$ and $\mathcal{D}$; the *similarity surfaces* images (of size $dim_r \times dim_c$) are a raster representation of $s(\mathbf{t})$.

Again, a rectangular shape is drawn around each best match, excluding the stereo match example, where the full disparity map is shown.

5.5.1 The Hausdorff method as a local correlation method

The proposed method can be used as a correlation method to find correspondences in stereo pairs and motion sequences.

Figure 5.8 shows a well-known stereo pair with the corresponding disparity map. To compute the disparity map of the left image, the correspondence method is applied at each pixel of the left image: the model is a neighbourhood of it (the size of which, is usually between 3×3 and 7×7), while the image is a subset of the right image, selected with the help of the epipolar constraint. To obtain the right disparity map, it suffices to swap left and right in the above description. Left-right consistency is performed, followed by a post-processing to fill the holes caused by multiple matchings and occlusions. If we choose very small dilations (close to zero), the results of our method are comparable to the ones of the SSD; if we increase the dilation, we take better care of occlusions and bound-

Figure 5.7: A stereo pair with its disparity map.

aries but we lose in terms of localization. On small areas as the neighbourhood of pixels, methods based on local ordering [BN98, ZW94] or robust statistics [BBW88, MMRK91] are more effective. Our method, instead, is much better suited for application to larger image regions: we have used it successfully to track small image regions through an image sequence. The model has been selected from the first frame and then matched along the sequence: Figure 5.9 shows a search in an image sequence (the figure contains frames 0, 20, 30, 40). A detail (the head of the statue) has been cropped from the first image, and sought in all the images of the sequence. During the image acquisition, the camera rotated around a statue, inducing shape deformations (a dilation in all the three directions had to be applied) and occlusions. The binary maps in the middle row represent the positive scores distribution. The bottom row shows the results obtained with SSD on the same input. Here and in the following examples it will be shown that wherever the search for the best match is ambiguous (more than one match is possible) SSD achieves poorer results. This is due to the fact that the SSD similarity surface is smoother and therefore it is more difficult to locate its maxima. See Figure 5.10 for a comparison of the similarity surfaces obtained with the two methods.

In Figure 5.10, both rows show, from left to right, the model, the positive scores distribution (which can only be computed with our method), the result obtained with our method (top) and SSD (bottom), and the similarity surface. The poor solution found with SSD is due to the smoothness of the similarity surface: the figure shows that high values are scored in wide areas of the image (the surface is brighter where the computed similarity

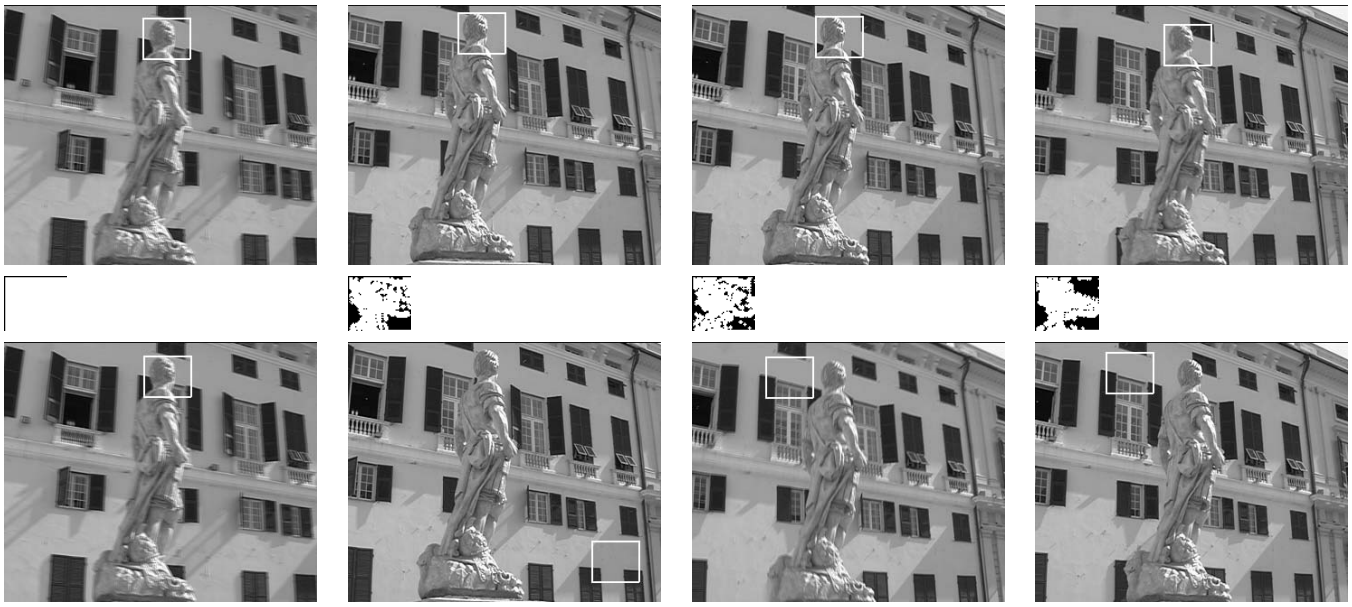


Figure 5.8: Samples of search through an image sequence. Top row: results obtained with our method on frames 0, 20, 30, 40. Middle row: positive scores distributions (see text). Bottom row: results obtained with SSD on the same frames.

is higher), therefore the localization of the best match is not reliable (note, instead, the white spot which is a well defined maximum in our similarity surface).

5.5.2 The Hausdorff method for object identification

Figure 5.11 shows, on the top left, a reference image used as a model in a set of object identification experiments. Results are shown in the two right-most column of Figure 5.11. These results fully represent the resistance of the method to occlusions and small scale changes. As usual, the best match areas are surrounded by a rectangular shape, but in this case, the brightness of the rectangle is proportional to the value of the match.
In Figure 5.12 we present a few frames selected from the same collection of images together with their positive scores distribution: this is another interesting example of how, in most cases, the positive scores distribution gives an immediate feedback of the location of occlusions and scene changes.

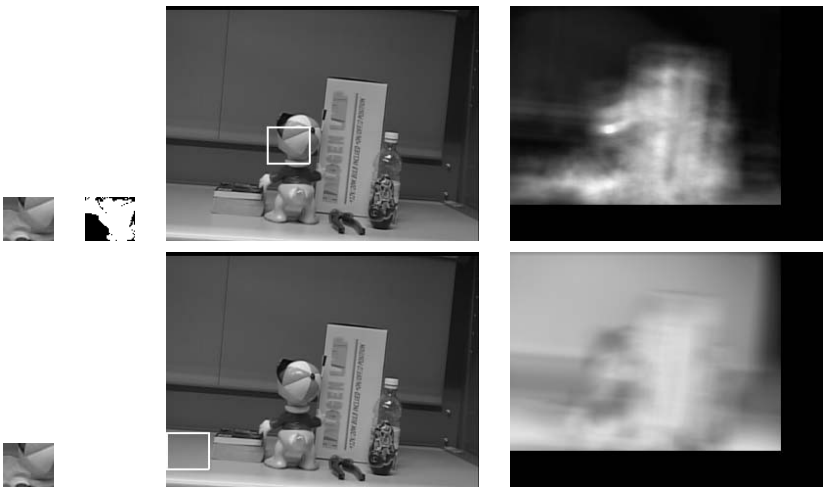


Figure 5.9: Example local search in presence of occlusion. Top row (our method): the model, the similarity scores, the result, the similarity surface. Top row (SSD): the (same) model, the result, the similarity surface.

5.5.3 The influence of thresholds

In the case of the grey-level similarity method presented in this chapter, the results are influenced by two thresholds: the first one is the degree of similarity (i.e., the dilation, the maximum distance allowed between two images), the second one is the fraction of the pixels which passed the similarity test.

Whenever a method depends on variable thresholds, it is interesting to study the influence they have on the results obtained. For simplicity the experiment is described on a specific example: the search of the model shown in Figure 5.13 in the image sequence, of which a few frames are shown in Figure 5.14.

We carry out this type of experiment in the following way: after choosing the model, we analyze manually the data and count the occurrences of the model inside the test images. In this sequence there is exactly one occurrence of the statue in each image, thus the ideal number of *positive results* is 19. However, as Figure 5.14 shows clearly, from the first image to the last one there is a large variation of the viewing position, therefore it is not clear if the model chosen is a good representation of the statue appearence in the last part of the sequence; this shows clearly how the manual operation of counting the positives, is actually rather subjective, but, on the other hand, it represents faithfully the

Figure 5.10: On the top left the model. On the central and right column a few results of object identification. We tested our method by allowing small illumination and scale changes (the last two images were acquired a few days after the acquisition of the model), occlusions, and small pose changes.

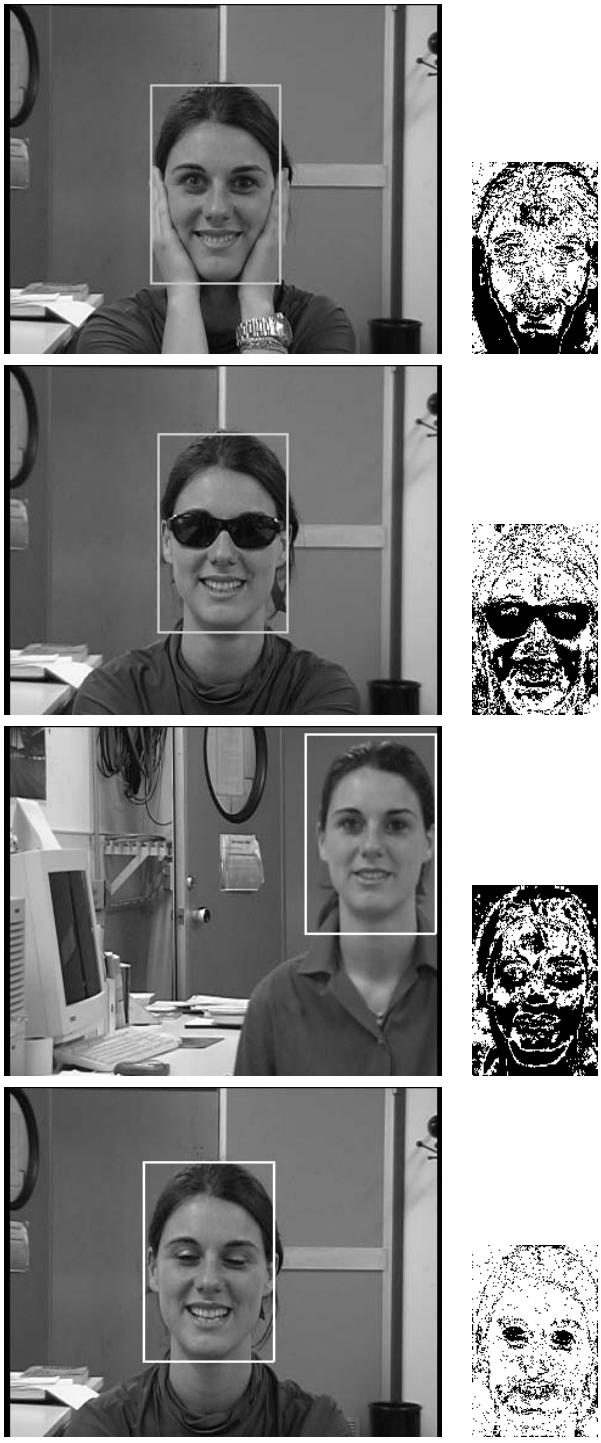


Figure 5.11: Other examples of object identification from the same images collection of Figure 5.11. The model is still the one shown at the top left of Figure 5.11. This time we include the positive scores distribution, to show how they can be useful to locate occlusions.

Figure 5.12: The model searched for in the sequence of Figure 5.14.

Figure 5.13: Image sequence where the search of the model (Figure 5.13) is performed. Notice that from the first frame (top left) to the last (bottom right) the angle of view has changed entirely, therefore the model chosen is not necessarily a good representation of the last images.

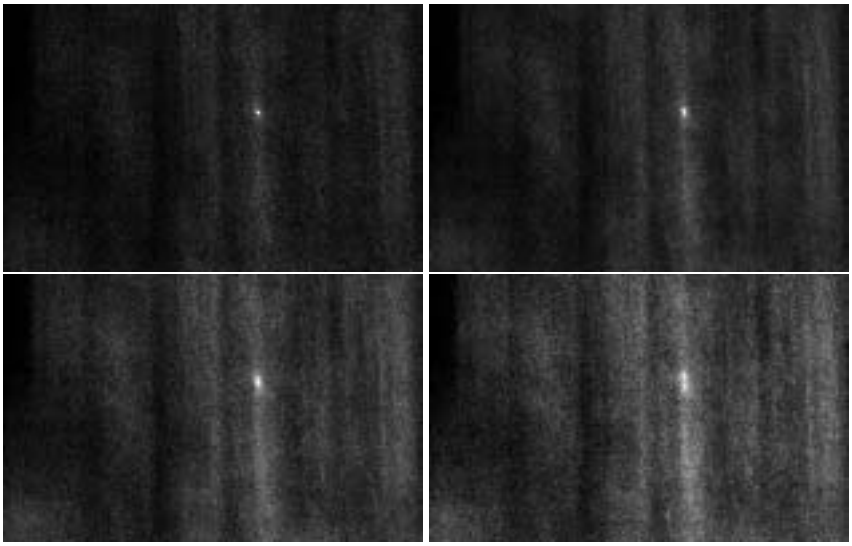


Figure 5.14: Examples of similarity surfaces resulting from the comparison of a model at four different scales with an image

actual problem of deciding when an identification system performs well or not. Usually, some degree of subjectivity is allowed: the only important thing is to be consistent in interpreting the results obtained.

Going back to our experiment, we perform a search of the model, for each image, at different scales (in this case the model was rescaled in 10 possible sizes). Each search produces a similarity surface: we rescale all the similarity surfaces related to the same test image to the same scale, and merge the results (see Figure 5.15): a position, (i, j), contains a positive match if at least one of the surfaces indicates a positive match in that position. To avoid spurious positive or negative matches, we subsample the search area over the surfaces in 10×10 regions. Each region would produce a positive match if it contained *at least* a positive one, a negative match otherwise. The performance of the search is evaluated by varying the percentage of positives required to a potential match to be positive, it can be represented with ROC curves [4].

Figure 5.16 compares different ROC curves obtained with different dilations. On the x

[4]Receiver Operating Characteristic (ROC) curves will be described more in detail in Chapter 7. ROC curves are a good representation because they describe the performances of a system indipendently from the threshold used to build the curve. Each point of an ROC curve represents a pair of false-alarms and hit-rate of the system, for a different threshold. In this case the threshold is given by the percentage of area which needs to match for an image region to be a positive match.

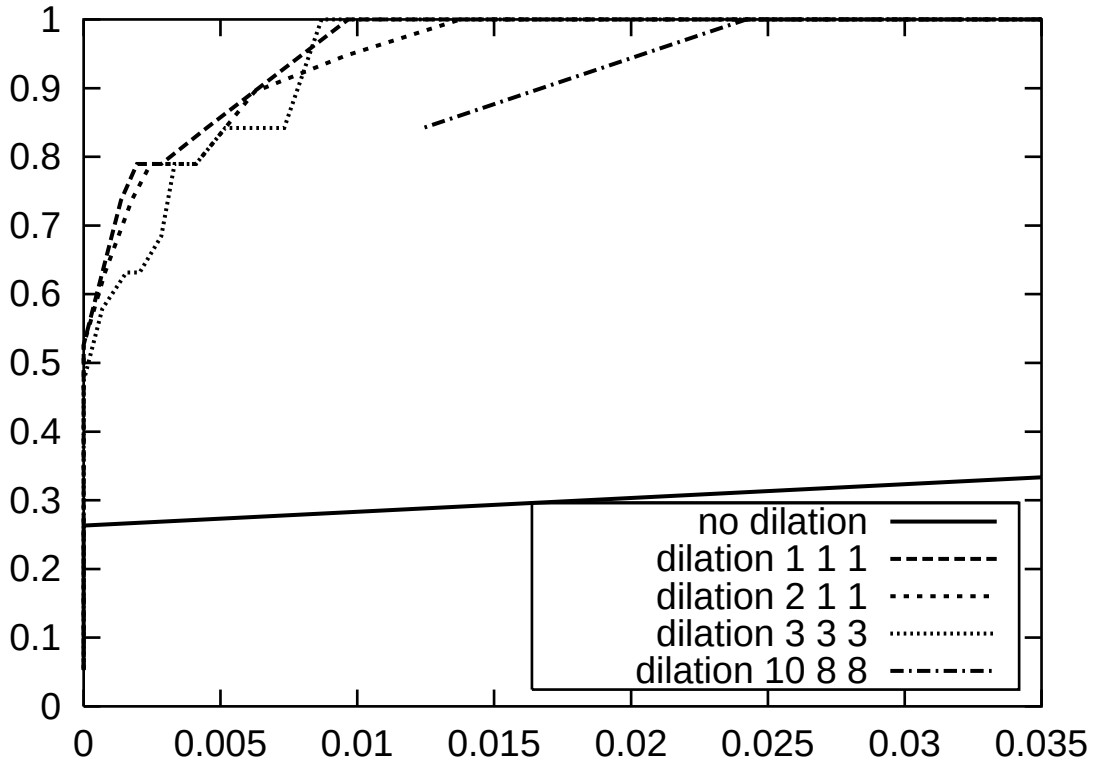


Figure 5.15: R.O.C. curves obtained by applying different dilations.

axis is the false positive rate, on the y axis the hit rate. The better are the performances of a system the quicker its ROC curve will grow. From this plot it is clear that adding a dilation, even minimal, the results improve substantially. On the contrary if we choose a substantial dilation to be sure to detect all the positives, we will reach more slowly the 100% hit rate. Intermediate dilations do not seem to affect the results in significant ways.

Discussion of Part II

Part II was devoted to studying the similarity between grey-level images — a first step towards image-based object representation and identification. Since we described a similarity method based on the Hausdorff distance, Part II started with an introductory chapter to the Hausdorff distances and their properties, and continued with a chapter entirely devoted to the review of correlation-based similarity measures, to introducing our similarity measure and its extensions, and to demonstrating the potential of this method on a variety of examples.

The method we introduced presents many interesting aspects: on one hand, it was demonstrated to tolerate small scale and illumination changes and geometric detormations, and on the other, it can be implemented efficiently and can easily be extended to multiscale.

It can be applied as it is for *brute-force* object identification based on image models, in applications where illumination and appearence changes of the objects from the models can be controlled or at least monitored (we are currently using it in a prototype for face identification of a limited number of people). It can also be used as a similarity function within more complicated object recognition systems, such as the learning methods described in the Part III of this thesis: in Chapter 7 we will describe an example of such a use.

Part III

3D object representation and identification

Outline of Part III

Object detection and classification have often been addressed in computer vision by the *learning from examples* paradigm. A classic approach is the one of finding separating functions which distinguish between positive and negative examples of the object we aim at representing. One alternative choice, which is particularly appealing in the cases when negative examples are not easy to characterize, is to design systems which learn only positive examples.

A kernel method to accomplish this goal consists of a *representation stage* — which computes the smallest sphere in feature space enclosing the positive examples — and a *classification stage* — which uses the obtained sphere as a decision surface to determine the positivity of new examples.

In the case the positive examples are different representative views of a 3D object, the representation stage can be seen as effectively the modelling of a 3D object. This model can be used as a description of the object which allows us to detect it or recognize it in single novel views.

In this last part of the thesis, after an introduction to kernel methods, we describe a kernel-based method to learn one class at a time. In particular, the main contribution of this part is the introduction of a kernel for grey-level images, well suited to representing, identifying, and recognizing 3D objects from unconstrained images.

Chapter 6 reviews kernel methods, which represent a means to increase the classification power of linear learning methods by projecting the data into a high dimensional feature space where a linear machine can be used; in the final part of the chapter we give a brief

overview of Support Vector Machines, the best known of kernel methods, and (for the case of classification problems only) describe a SV method for learning one class at a time.

Chapter 7 presents the problem of designing kernels for images. The main contribution of this chapter is the introduction of a function based on the similarity measure presented in Chapter 5. This function, although it is not a Mercer's kernel, can still be used as a kernel in the case of one class learning. This issue is discussed and the effectiveness of this function as a kernel for novelty detection is demonstrated on several data sets of faces and of 3D objects of artistic relevance (statues). Also, a variant of the function (based on a careful definition of the dilation) with the properties of a Mercer's kernel is introduced.

Chapter 6

Kernel-based methods for statistical learning

Kernel methods have recently, since the influential work of Vapnik [Vap95, Vap98], attracted increasing attention. They reduce a learning problem of classification or regression to a multivariate function approximation problem in which the solution is found as a linear combination of certain positive definite functions, called kernels, centered at the examples [EPP00a, GJP95, Wah90]. In this chapter, we first provide an introduction to kernel methods, and then describe briefly a well known technique based on kernels, Support Vector Machines [Vap98, CST00], concentrating on the case of binary classification problems[1]*. Finally, we introduce a SV method for learning one class at the time [Vap95, SBV95, TD99, CB01] inspired by SVM, which can be used as an alternative to finding negative examples for a binary classification problem.*

6.1 Introduction to linear classification

The traditional approach to programming is based on instructing a computer in order to accomplish the desired task. It implies that the programmer knows a "recipe" for solving the problem, and translates it into a language that the machine can understand and apply

[1] *The interested reader may refer to the seminal works of Vapnik [Vap95, Vap98], the book by Cristianini and Shawe-Taylor [CST00] which we use as the main guide for this introduction, or to the tutorial by Burges [Bur98].*

to the input data. There are, however, cases when this recipe is not known in detail, or is difficult to express in simple machine readable instructions; in other cases there are sometimes too many exceptions to the main solution for it to be useful. In all these situations an alternative is to "teach" the computer the input/output relationship, using examples. This philosophy is known as the *learning approach*. In the case that the system is taught by a set of input/output pairs it is referred to as *supervised learning* (*unsupervised learning*, which we will not discuss in this thesis, treats the case when there are no output data, and the learning task is to discover some properties of the process which generated the data).

In supervised learning, the learning machine is given a training set, defined as follows:

Definition 6.1.1 (training set) We denote the input space by X and the output domain by Y. Usually $X \subseteq \mathbb{R}^n$, while for binary classification, $Y = \{-1, 1\}$. A *training set* is a collection of training examples, and is usually denoted by

$$S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\} \in (X \times Y),$$

where ℓ is the number of examples. The $\mathbf{x}_i$ will be referred to as the examples or *instances*, while the y_i will be referred to as the *labels*.

At this point, one could choose among various sets of hypothesis for the classification problem. Among the possibilities, *linear functions* are the best understood and the simplest to apply: traditional statistics (with the linear discriminant introduced by Fischer in 1936) and the classical neural network literature (with the Perceptron, studied in the sixties) present methods for using linear functions to discriminate between two classes.

Binary linear classification is frequently performed by using a function $f : X \subseteq \mathbb{R}^n \to \mathbb{R}$ in the following way: the input vector $\mathbf{x}$ is associated to a positive class if $f(\mathbf{x}) \geq 0$, and to a negative class otherwise. The function, f, chosen in the case of linear classification is of the form

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \mathbf{x} \rangle + b$$

where $\mathbf{w}$ and b are the control parameters of the linear function. The *decision rule* is

given by sign($f(\mathbf{x})$). The learning methodology implies that these parameters must be learned from the data. In general, real applications are too complex to be formalized as linear classification problems — multi-layer neural networks (based on multiple layers of thresholded linear functions) were one of the first proposed solutions to this difficulty.
One important aspect of most linear machines is the fact that they can be expressed as their *dual representation*[2]:

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i y_i \langle \mathbf{x}_i \cdot \mathbf{x} \rangle + b. \tag{6.1}$$

Duality is a crucial property that allows the introduction of kernel methods. In the dual representation the training data never appear isolated, but are always as entries of the Gram matrix [3]. Also, in the dual representation of the decision function, only the inner products between input data and the test point are needed.

6.2 Implicit mapping into feature space

Kernel methods offer an alternative to linear classification, based on projecting the data into a high-dimensional *feature space* to increase the range of problems solvable by linear learning machines.
In order to learn non-linear relations with a linear machine, we need to apply a non-linear mapping of the data to a *feature space*, in which the linear machine can be used. Given an input vector $\mathbf{x}$ in the input space X, it is mapped into the feature space $F = \{\boldsymbol{\phi}(\mathbf{x}) | \mathbf{x} \in X\}$:

$$\mathbf{x} = (x_1, \ldots, x_n) \rightarrow \boldsymbol{\phi}(\mathbf{x}) = (\phi_1(\mathbf{x}), \ldots, \phi_N(\mathbf{x})).$$

Figure 6.1 shows an example of a feature mapping from a two dimensional input space to a two dimensional feature space, where the data cannot be separated by a linear function in the input space, but they can in the feature space.

[2]The dual representation, obtained in different ways for different classificators, can always be written in the general form of Equation (6.1). In Section 6.3 we will show how to obtain the dual representation for the case of Support Vector Machines.

[3]Given a set of vectors $\{\mathbf{x}_1, \ldots \mathbf{x}_N\}$ of an inner product space, the Gram matrix is defined as $\mathbf{G} = (\langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle)_{i,j=1}^{\ell}$ and it is also referred to as *kernel matrix*.

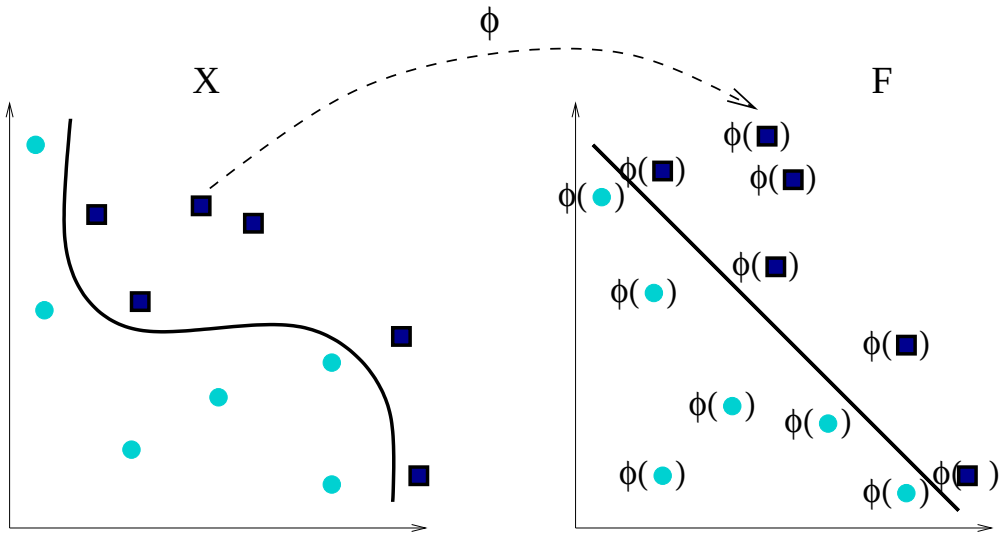


Figure 6.1: A mapping from the input space to the feature space, which allows a linear classification.

Thus, the set of hypotheses considered will be functions of the type:

$$f(\mathbf{x}) = \langle \mathbf{w} \cdot \boldsymbol{\phi}(\mathbf{x}) \rangle + b$$

where $\boldsymbol{\phi} : X \to F$ is a non-linear map from the input space to some feature space F.

Finding a suitable representation of the data (i.e., choosing the feature space) is one of the most delicate steps of this approach to learning: ideally one should choose a representation that suits the specific learning problem.

The key feature of kernel-based learning is that the mapping, ϕ, into the high-dimensional feature space need not be calculated explicitly, thanks to the characteristics of the dual representation discussed in the previous section. This is rather fortunate if one considers that, to build explicitly a polynomial of degree 4 or 5 in a 200 dimensional input space, one must construct hyperplanes in a billion dimensional feature space. This crucial property is based on the fact that the only quantities in the feature space that one needs to compute are scalar products of the form $\langle \boldsymbol{\phi}(\mathbf{x}) \cdot \boldsymbol{\phi}(\mathbf{y}) \rangle$, hence the full knowledge of $\boldsymbol{\phi}$ is not necessary.

One common strategy, therefore, is to compute the data representation by means of an implicit mapping, which means that neither the feature space nor the mapping are spec-

ified explicitly. The dual representation enables a hypothesis to be expressed as a linear combination of the training points, so that the decision rule can be evaluated using just inner products between the test point and the training points:

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i y_i \langle \boldsymbol{\phi}(\mathbf{x}_i) \cdot \boldsymbol{\phi}(\mathbf{x}) \rangle + b.$$

If we have some way of computing the inner product $\langle \boldsymbol{\phi}(\mathbf{x}_i) \cdot \boldsymbol{\phi}(\mathbf{x}) \rangle$ in feature space directly as a function of the original input points, it becomes possible to merge the mapping into the feature space with the use of the linear machine. This direct computation method is called a *kernel* function.

Definition 6.2.1 A *kernel* [4] is a symmetric function K, such that for all $\mathbf{x}, \mathbf{z} \in X$

$$K(\mathbf{x}, \mathbf{z}) = \langle \boldsymbol{\phi}(\mathbf{x}) \cdot \boldsymbol{\phi}(\mathbf{z}) \rangle,$$

where $\boldsymbol{\phi}$ is a mapping from X to a feature space with an inner product.

Once the kernel function is known, the decision rule can be evaluated using at most ℓ evaluations of the kernel:

$$f(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b. \tag{6.2}$$

An important consequence of this is that the learning algorithms and the theory behind them can largely be decoupled from the specific application areas. The latter can be encoded into the design of appropriate kernel functions, thus kernel engineering is a central issue in kernel-based methods for machine learning.

6.2.1 Making kernels

The use of the "kernel trick" allows us to define the kernel function directly, and, in doing so, to define implicitly a mapping between the input space and feature space. Indeed, as

[4] For a clear introduction to kernel functions the reader may refer to [CST00].

pointed out in [CST00], "defining a kernel function for an input space is frequently more natural than creating a complicated feature space".
To use this, we must define the properties which characterize a well-defined kernel, i.e., a function $K(\mathbf{x}, \mathbf{z})$ which is an inner product in some feature space.

Definition 6.2.2 A function $K : X \times X \to \mathbb{R}$ is *positive definite* if, for all n and all possible choices of $\mathbf{x}_1, \ldots, \mathbf{x}_n \in X$ and $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0. \tag{6.3}$$

Equation (6.3) says that all matrices $\mathbf{K}$ of all orders built from a positive definite function K are semi-definite positive.
A theorem of functional analysis due to Mercer [CH62] allows us to write a positive definite function as an expansion of certain functions $\phi_k(\mathbf{x})$, $k = 1, \ldots, N$, with N a possibly infinite number:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{N} \lambda_K \phi_k(\mathbf{x}_i) \phi_k(\mathbf{x}_j) = \langle \boldsymbol{\phi}(\mathbf{x}_i) \cdot \boldsymbol{\phi}(\mathbf{x}_j) \rangle.$$

For a rigorous statement of Mercer's theorem, the interested reader is referred to [CH62].
A positive definite function is also called a *Mercer's kernel*, or simply a *kernel*.
Whether or not this espansion is explicitly computed, it suffices to note that for any kernel K, $K(\mathbf{x}_i, \mathbf{x}_j)$ can be thought of as the inner product between the vectors $\boldsymbol{\phi}(\mathbf{x}_i) = (\phi_1(\mathbf{x}_i), \ldots, \phi_N(\mathbf{x}_i))$ and $\boldsymbol{\phi}(\mathbf{x}_j) = (\phi_1(\mathbf{x}_j), \ldots, \phi_N(\mathbf{x}_j))$.
We conclude this section with examples of kernel functions.

Linear kernel

The inner product

$$K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x} \cdot \mathbf{z} \rangle \tag{6.4}$$

provides an example of a kernel, which corresponds to the identity map from input space to feature space. The separating surface in input space, therefore, is a hyperplane.

Polynomial kernels

A simple example of a non-linear mapping is given by:

$$K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x} \cdot \mathbf{z} \rangle^2.$$

This is equivalent to an inner product between feature vectors, with

$$\boldsymbol{\phi}(\mathbf{x}) = (x_i x_j) \quad i, j = \{1, \ldots, n\}.$$

Indeed

$$\begin{aligned} \langle \mathbf{x} \cdot \mathbf{z} \rangle^2 &= \left(\sum_{i=1}^{n} x_i z_i \right)^2 = \left(\sum_{i=1}^{n} x_i z_i \right) \left(\sum_{j=1}^{n} x_j z_j \right) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j z_i z_j = \sum_{i=1}^{n} \sum_{j=1}^{n} (x_i x_j)(z_i z_j), \end{aligned}$$

Here the features are all monomials of degree 2. A more general feature space can be obtained by considering the kernel

$$\begin{aligned} (\langle \mathbf{x} \cdot \mathbf{z} \rangle + c)^2 &= \left(\sum_{i=1}^{n} x_i z_i + c \right) \left(\sum_{j=1}^{n} x_j z_j + c \right) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j z_i z_j + 2c \sum_{i=1}^{n} x_i z_i + c^2 \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n} (x_i x_j)(z_i z_j) + \sum_{i=1}^{n} \sum_{j=1}^{n} (\sqrt{2c} x_i)(\sqrt{2c} z_i) + c^2, \end{aligned}$$

whose features are all monomials of degree up to 2, with the relative weights between the degree 1 and 2 controlled by the parameter c.

A similar analysis can be given for the kernels

$$K(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x} \cdot \mathbf{z} \rangle^d, \tag{6.5}$$

and

$$K(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x} \cdot \mathbf{z} \rangle + c)^d \tag{6.6}$$

for $d \geq 2$. In the first case the $\binom{n+d-1}{d}$ distinct features are all the monomials of degree d, and in the second there are $\binom{n+d}{d}$ distinct features corresponding to all monomials with degree up to d. The separating function in the input space corresponding to a hyperplane in such feature space is a polynomial curve of degree d, so these kernels are usually called *polynomial kernels.*

Gaussian kernel

The function

$$K(\mathbf{x}, \mathbf{z}) = \exp(-||\mathbf{x} - \mathbf{z}||^2/\sigma^2), \tag{6.7}$$

for some $\sigma \in \mathbb{R}$ defines a Gaussian kernel. A learning machine equipped with a Gaussian kernel belong to the class of Radial Basis Functions classifiers, where the centers are the support vectors and the coefficients are the Lagrange multipliers (except for the sign).

6.2.2 Making kernels from kernels

In the this section we list a number of properties, which allow us to create more complicated kernels. The proofs of these properties can be found in [CST00].
Let K_1 and K_2 be kernels over $X \times X$, $X \subset \mathbb{R}^n$ and $a \in \mathbb{R}^+$, then the following functions, K, are also kernels:

$$K(\mathbf{x}, \mathbf{z}) = K_1(\mathbf{x}, \mathbf{z}) + K_2(\mathbf{x}, \mathbf{z}), \tag{6.8}$$

$$K(\mathbf{x}, \mathbf{z}) = aK_1(\mathbf{x}, \mathbf{z}), \tag{6.9}$$

$$K(\mathbf{x}, \mathbf{z}) = K_1(\mathbf{x}, \mathbf{z})K_2(\mathbf{x}, \mathbf{z}). \tag{6.10}$$

Let $f(\cdot)$ be a real valued function on X, then we also have

$$K(\mathbf{x}, \mathbf{z}) = f(\mathbf{x})f(\mathbf{z}). \tag{6.11}$$

If $\boldsymbol{\phi} : X \to \mathbb{R}^m$, and K_3 is a kernel over $\mathbb{R}^m \times \mathbb{R}^m$

$$K(\mathbf{x}, \mathbf{z}) = K_3(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{z})). \tag{6.12}$$

If $\mathbf{B}$ is a positive semi-definite $n \times n$ matrix, then

$$K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^\top \mathbf{B}\mathbf{z}. \tag{6.13}$$

Let $K_1(\mathbf{x}, \mathbf{z})$ be a kernel over $X \times X$, $\mathbf{x}, \mathbf{z} \in X$, $p(x)$ a polynomial with positive coefficients, then the following functions are also kernels:

$$K(\mathbf{x}, \mathbf{z}) = p(K\mathbf{x}, \mathbf{z})), \tag{6.14}$$

$$K(\mathbf{x}, \mathbf{z}) = \exp(K\mathbf{x}, \mathbf{z})), \tag{6.15}$$

Notice that the difference of two kernels is not necessarily a kernel.

6.3 Support Vector Machines

Support Vector Machines (SVMs) are the most known class of algorithms which use the idea of kernel substitution. In this section we briefly overview them concentrating on SVMs for binary classification (SVMC).

We start by recalling that the ability of a hypothesis to classify correctly previously unseen data is known as *generalization*. From the point of view of statistical learning theory the motivation for considering binary SVMs classifiers comes from the theoretical bounds on the generalization error [Vap98]. We do not quote the relevant theorem here — but remind the interested reader to [Vap98] — but just note two important aspects: the first one, it that the upper bound on the generalization error does not depend on the dimension of

the space, the second one that the error bound is minimised by maximizing the minimal distance between the separating hyperplane and the closest datapoints to the hyperplane.

6.3.1 Hard margin optimization

Let us assume that we are given a set S of points $\mathbf{x}_i \in \mathbb{R}^n$ with $i = 1, 2, \ldots, \ell$. Each point $\mathbf{x}_i$ belongs to either of two classes and thus is given a label $y_i \in \{-1, 1\}$. The goal is to establish the equation of a hyperplane that divides S leaving all the points of the same class on the same side and maximizing the minimum distance between either of the two classes and the hyperplane.

The set S is *linearly separable* if there exists $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ so that

$$y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad \text{for } i = 1, 2, \ldots, \ell. \tag{6.16}$$

The hypothesis space in this case is the following set of functions:

$$f(\mathbf{x}) = \text{sign}(\langle \mathbf{w} \cdot \mathbf{x} \rangle + b) \tag{6.17}$$

The pair $(\mathbf{w}, b)$ defines a hyperplane of equation

$$\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0 \tag{6.18}$$

named *separating hyperplane*.

The signed distance d_i of a point $\mathbf{x}_i$ from the separating hyperplane $(\mathbf{w}, b)$ is given by

$$d_i = \frac{\langle \mathbf{w} \cdot \mathbf{x} \rangle + b}{||\mathbf{w}||} \tag{6.19}$$

Notice that if the parameters $\mathbf{w}$ and b are scaled by the same quantity, the decision surface given by Equation (6.18) is unchanged. In order to remove this redundancy a *canonical representation* is chosen for each decision surface, that is a one-to-one correspondence between separating hyperplanes and their parametric representation. The canonical rep-

resentation of a separating hyperplane described by the equation (6.18) is obtained by rescaling the pair $(\mathbf{w}, b)$ into the pair $(\mathbf{w}', b')$, in a way that the distance of the closest point is equal to $1/||\mathbf{w}||'$:

$$\min_{i=1,\ldots,l} y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) = 1. \tag{6.20}$$

Given a linearly separable set S, the *Optimal Separating Hyperplane* (OSH) is the separating hyperplane which maximize the distance to the closest point of S, without any classification error. That is, the OSH is the hyperplane that satisfies Equation (6.16), while minimizing w.

Finding the Optimal Separating Hyperplane

Formalizing what has been said in the previous section, to construct the OSH we need to solve the following Quadratic Problem (QP) [BS79]:

$$\begin{array}{lll} \textbf{Problem CL1:} & \min_{\mathbf{w},b} & \Phi(\mathbf{w}) = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle \\ & \text{subject to} & y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1, \quad i = 1, 2, \ldots, \ell \end{array} \tag{6.21}$$

The quantity $2/||\mathbf{w}||$ is named *margin* and it measures the distance between the two classes in the direction of $\mathbf{w}$. The OSH can also be described as the separating hyperplane that maximizes the margin.

Problem **CL1** can be solved using a standard QP optimization technique[BS79].

This optimization problem can be transformed into its dual problem, and solved with the technique of Lagrange multipliers[BS79].

We first construct the primal Lagrangian. If we denote with $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_\ell)$ the ℓ non-negative Lagrange multipliers associated with the constraints of the problem (6.21), then the solution of the problem is determined by determining the *saddle point* of the Lagrangian:

$$L = L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{\ell} \alpha_i [y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) - 1]. \tag{6.22}$$

At the saddle point, L has a minimum for $\mathbf{w} = \mathbf{w}^*$ and $b = b^*$, and a maximum for $\boldsymbol{\alpha} = \boldsymbol{\alpha}^*$.

Differentiating (6.22) and setting the results equal to zero, we obtain:

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{\ell} y_i \alpha_i = 0, \tag{6.23}$$

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i = 0 \tag{6.24}$$

where $\frac{\partial L}{\partial \mathbf{w}} = \left(\frac{\partial L}{\partial w_1}, \ldots, \frac{\partial L}{\partial w_\ell}\right)$. Problem **CL1** reduces to the constrained maximization of the function:

$$\mathcal{L}(\boldsymbol{\alpha}) = \sum_{i=0}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle. \tag{6.25}$$

A precise formulation of the dual problem is the following:

$$\begin{array}{lll} \textbf{Problem DCL1:} & \max_{\boldsymbol{\alpha}} & -\frac{1}{2}\alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle + \sum \alpha_i \\ & \text{subject to} & \sum y_i \alpha_i = 0 \\ & & \boldsymbol{\alpha} \geq 0, \end{array} \tag{6.26}$$

Note that, as pointed out in Section 6.2, in the dual representation of the problem, the input data never appear isolated, but always in the form of an inner product between pairs of instances. Thus we can map the data into an alternative high-dimensional space, by means of a kernel function appropriate for the specific learning problem. In general, the objective function of Problem **DCL1** can be rewritten as:

$$\max_{\boldsymbol{\alpha}} \; -\frac{1}{2}\alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) + \sum \alpha_i.$$

Support vectors

Among the Kuhn-Tucker conditions for Problem **CL1**, the following is verified[5]

[5]The full Kuhn-Tucker conditions, for the solution x_0 of the minimization problem

$$\begin{array}{lll} \min_x & f(x) & \\ \text{subject to} & g_k(x) \geq 0 & k = 1, 2, \ldots, K \\ & h_j(x) = 0 & j = 1, 2, \ldots, J \end{array}$$

where the Lagrangian function is $L(x, u, v) = f(x) - u_k g_K(x) - v_j h_j(x)$, are:

$$\alpha_i^* \left(y_i(\langle \mathbf{w}^* \cdot \mathbf{x}_i \rangle + b^*) - 1 \right) = 0, \quad i = 1, 2, \ldots, \ell. \tag{6.27}$$

From Equation (6.27) one can note that the only α_i^* that can be non-zero, are those for which the constraints (6.16) are satisfied with the equality sign.

The corresponding points $\mathbf{x}_i$ are termed *support vectors* and they are the points of S closest to the OSH.

The decision function is based on the sign of:

$$f(\mathbf{z}) = \sum_{i=1}^{\ell} y_i \alpha_i K(\mathbf{z}, \mathbf{x}_i) + b,$$

6.3.2 Soft margin optimization

In general, in real-world problems, it is not possible to use maximal margin classifiers, unless we use very powerful kernels, and cause data overfitting. The main problem of using maximal margin classifiers, is that they are based on a really strong hypothesis, that is, they assume that the training set does not contain errors. In practice this in often not the case.

A more realistic setting, is the following: we are looking for a linear separating surface, but we are not sure that all the constraints of Problem **CL1** are satisfied. In this case, the previous analysis can be generalized by introducing a new set of ℓ non negative variables, $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_\ell)$ that measure the amount of violation of the constraint[CV95]:

$$y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \quad i = 1, 2, \ldots, \ell. \tag{6.28}$$

The presence of the variables ξ_i allows a number of misclassified points. If the point $\mathbf{x}_i$

1. $\nabla f(x_0) - \sum u_k \nabla g_k(x_0) - \sum v_j \nabla h_j(x_0) = 0$
2. $g_k(x_0) \geq 0 \quad \forall k$
3. $h_j(x_0) \geq 0 \quad \forall j$
4. $u_k g_k(x_0) = 0 \quad \forall k$
5. $u_k \geq 0 \quad \forall k$

satisfies the constraint of Problem **CL1**, then $\xi_i = 0$ and the constraint (6.28) reduces to (6.16). Formally, to obtain the generalized OSH, one solves the following problem:

$$\begin{array}{lll} \textbf{Problem CL2:} & \min_{\mathbf{w},b} & \Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}\cdot\mathbf{w} + C(\sum \xi_i) \\ & \text{subject to} & y_i(\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b) \geq 1 - \xi_i, \qquad i = 1, 2, \ldots, \ell \\ & & \xi_{\mathbf{i}} \geq 0 \qquad\qquad\qquad\qquad\qquad\qquad . \end{array} \tag{6.29}$$

The parameter C can be considered as a regularization parameter. $C(\sum \xi_i)$ keeps under control the number of misclassified points. Other monotonic convex functions have been tried [CV95]. In analogy with what was done for the separable case, Problem **CL2** can be transformed into the *dual*:

$$L = L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\langle \mathbf{w}\cdot\mathbf{w}\rangle + C\sum_{i=1}^{\ell}\xi_i - \sum_{i=1}^{\ell}\alpha_i[y_i(\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b) - 1 + \xi_i] - \sum_{i=1}^{\ell}\beta_i\xi_i. \tag{6.30}$$

Differentiating (6.30) and setting the results equal to zero, we obtain:

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{\ell} y_i\alpha_i = 0, \tag{6.31}$$

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{\ell}\alpha_i y_i \mathbf{x}_i = 0, \tag{6.32}$$

$$\frac{\partial L}{\partial \boldsymbol{\xi}} = C - \alpha_i - \beta_i = 0 \tag{6.33}$$

Similarly to what it has been done in the separable case, with the use of the "kernel trick", Problem **CL2** reduces to the following problem

$$\begin{array}{lll} \textbf{Problem DCL2:} & \max_{\boldsymbol{\alpha}} & -\frac{1}{2}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) + \sum \alpha_i \\ & \text{subject to} & \sum y_i\alpha_i = 0 \\ & & 0 \leq \alpha_i \leq C, \qquad\qquad i = 1, 2, \ldots, \ell. \end{array} \tag{6.34}$$

The new Kuhn-Tucker conditions are

$$\alpha_i^*(y_i(\langle \mathbf{w}^* \cdot \mathbf{x}_i \rangle + b^*) - 1 + \xi_i^*) = 0 \quad (6.35)$$

$$(C - \alpha_i^*)\xi_i^* = 0 \quad (6.36)$$

where ξ_i^* are the values of ξ_i in the saddle point. The original form of (6.36) is $\beta_i^* \xi_i^* = 0$, but it has been modified according to (6.33).

Similarly to the separable case, the points $\mathbf{x}_i$ for which $\alpha_i > 0$ are called *support vectors*. Here we also have to distinguish between the support vectors for which $\alpha_i < C$ and those for which $\alpha_i = C$.

If $\alpha_i < C$ it follows (Equation (6.36)) that $\xi_i^* = 0$ and, then (Equation (6.35)) that these support vectors lie at a distance $1/w^*$ from the OSH. They are called *margin vectors*.

If $\alpha_i = C$, instead, the corresponding support vectors are either misclassified (if $\xi_i^* > 1$) or points correctly classified but closer than $1/w^*$ to the OSH (if $0 < \xi_i^* \leq 1$) or, even points lying on the margin (if $\xi_i^* = 0$). In any case we call them *errors*. The parameter C is meant to balance the empirical risk minimization and the minimization of the VC-dim. It is the only free parameter of SVMs. For this reason, studying the effect of small changes of C on the OSH, is relevant from both the theoretical and practical viewpoint[PV98].

6.4 Learning one class at a time

The approach of learning one class at a time is a natural choice for classification problems, whenever one class is easily characterized, while the other class is complementary ill defined. In object detection, where the object can be a face, a car, a specific person X, the class of positive examples is automatically understood from the problem statement, whereas it is not always clear, what sort of examples to choose to describe non-faces, non-cars, non-Xs. Among the numerous approaches to representing the class of negative examples, one possible alternative is to simply decide not to represent them! Instead of representing both classes, and then looking for a separation between the two of them, one could look for a separating function between the positive examples and “all the rest”. In the linear

case, instead of seeking the OSH, we estimate the sphere of minimum radius which contains "most" of the data of the training set. In the non-linear case, as for classification, the search of the sphere will be performed in the feature space.

This approach was first suggested in [Vap95] and [SBV95], and interpreted and used as a novelty detector[6] in [TD99] and [TYD99]. In this section we review the method described in [CB01] which we follow closely.

In the linear case, the primal minimization problem can be written as

$$\textbf{Problem ND1} \qquad \begin{array}{ll} \min_{R,\mathbf{x}_0,\boldsymbol{\xi}} & R^2 + C\sum_i \xi_i \\ \text{subject to} & (\mathbf{x}_i - \mathbf{x}_0)^2 \leq R^2 + \xi_i \\ & \xi_i \geq 0, \quad i = 1,\ldots,\ell \end{array} \tag{6.37}$$

with $\mathbf{x}_1, ..., \mathbf{x}_\ell$ the input data, R the sphere radius, $\mathbf{x}_0$ the sphere center. As for the binary case, the possible presence of outliers is countered by using slack variables, $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_\ell)$, $\xi_i \geq 0$, which allow for data points outside the sphere; C is a regularization parameter controlling the relative weight between the sphere radius and the outliers.

The dual problem can be obtained by the primal Lagrangian

$$L(R, \mathbf{x}_0, \xi, \boldsymbol{\alpha}, \boldsymbol{\beta}) = R^2 - \sum_i \alpha_i [R^2 - (\mathbf{x}_i - \mathbf{x}_0)^2 + \xi_i] - \sum_i \beta_i \xi_i + C \sum_i \xi_i \tag{6.38}$$

where α_i, β_i are the non-negative multipliers associated to the two sets of constraints of Problem ND1. The solution to this problem is determined by the saddle point of the Lagrangian, obtained by minimizing with respect to ξ_i, R, and $\mathbf{x}_0$ and maximizing with respect to α_i and β_i. By differentiating Equation (6.38) and setting the results equal to

[6] For this reason problems for which only positive examples are easily identifiable are sometimes referred to as *novelty detection* problems [TD99, CB01].

zero, we obtain

$$\begin{aligned}
\frac{\partial L}{\partial R} &= 2R - 2R\sum_{i=1}^{\ell}\alpha_i = 0 \\
\frac{\partial L}{\partial \mathbf{x}_0} &= 2\sum_{i=1}^{\ell}\alpha_i(\mathbf{x}_i - \mathbf{x}_0) = 0 \\
\frac{\partial L}{\partial \xi_i} &= -\alpha_i - \beta_i + C = 0 \qquad \forall i = 1, \ldots, \ell
\end{aligned}$$

yielding

$$\sum_{i=1}^{\ell}\alpha_i = 1, \tag{6.39}$$

$$\mathbf{x}_0 = \sum_{i=1}^{\ell}\alpha_i\mathbf{x}_i, \tag{6.40}$$

$$\beta_i = C - \alpha_i, \qquad \forall i = 1, \ldots, \ell \tag{6.41}$$

Using the Kuhn-Tucker conditions we have:

$$\begin{aligned}
\alpha_i[(\mathbf{x}_i - \mathbf{x}_0)^2 - R^2 - \xi_i] = 0 \\
(C - \alpha_i)\xi_i = 0
\end{aligned}$$

The dual formulation requires the solution of the QP problem

$$\begin{array}{ll}
\min_\alpha & -\sum_i \alpha_i\langle\mathbf{x}_i \cdot \mathbf{x}_i\rangle + \sum_i\sum_j \alpha_i\alpha_j\langle\mathbf{x}_i \cdot \mathbf{x}_j\rangle \\
\text{subject to} & \sum_i \alpha_i = 1 \\
& 0 \leq \alpha_i \leq C.
\end{array}$$

Notice that the sphere center $\mathbf{x}_0$ is a weighted sum of the examples expressed as $\mathbf{x}_0 = \sum_i \alpha_i\mathbf{x}_i$, while the radius R can be determined from the Kuhn-Tucker conditions associated to any unbounded training point — that is, any point of the training set for which $0 < \alpha < C$ — as $R^2 = (\mathbf{x} - \mathbf{x}_0)^2$. The training points for which $\alpha_i > 0$ are the *support vectors* for this learning problem.

The distance between a point and the center can be computed from the following equation

$$\begin{aligned} d(\mathbf{x})^2 &= (\mathbf{x}-\mathbf{x}_0)^2 = \\ &= \langle \mathbf{x}\cdot\mathbf{x}\rangle - 2\langle \mathbf{x}\cdot\mathbf{x}_0\rangle + \langle \mathbf{x}_0\cdot\mathbf{x}_0\rangle = \\ &= \langle \mathbf{x}\cdot\mathbf{x}\rangle - 2\sum_{i=1}^{\ell}\alpha_i\langle \mathbf{x}\cdot\mathbf{x}_i\rangle + \sum_{i,j=1}^{\ell}\alpha_i\alpha_j\langle \mathbf{x}_i\cdot\mathbf{x}_j\rangle \end{aligned}$$

In a similar way to the one described for Support Vector Machines, a non linear approach can be obtained with a mapping from the input space to a feature space. This can be done implicitly with a kernel function, as described in Section 6.1.

$$\begin{array}{lll} \textbf{Problem DND1} & \min_\alpha & -\sum_{i=1}^{\ell}\alpha_i K(\mathbf{x}_i,\mathbf{x}_i) + \sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j) \\ & \text{subject to} & \sum_{i=1}^{\ell}\alpha_i = 1 \\ & & 0 \le \alpha_i \le C. \end{array} \tag{6.42}$$

Using the kernel trick, the sphere center in feature space can not be computed explicitly, unless the mapping $\boldsymbol{\phi}$ to the feature space is known in an explicit form that allow the computation of $\boldsymbol{\phi}(\mathbf{x}_0) = \sum_i \alpha_i \boldsymbol{\phi}(\mathbf{x}_i)$. Instead, the distance $d_K(\mathbf{x})$ in feature space between the sphere center and a point $\mathbf{x}$ can be written as

$$[d_K(\mathbf{x})]^2 = K(\mathbf{x},\mathbf{x}) - 2\sum_{i=1}^{\ell}\alpha_i K(\mathbf{x},\mathbf{x}_i) + \sum_{i,j=1}^{\ell}\alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j). \tag{6.43}$$

As in the linear case, the radius R_K can be determined from the Kuhn-Tucker conditions associated to a support vector $\mathbf{x}$ for which $0 < \alpha < C$.
After training, a new data $\mathbf{z}$ is classified as a novelty if

$$K(\mathbf{x},\mathbf{x}) - 2\sum_{j=1}^{\ell}\alpha_j K(\mathbf{x},\mathbf{x}_j) + \sum_{j,k=1}^{\ell}\alpha_j\alpha_k K(\mathbf{x}_j,\mathbf{x}_k) = [d_K(\mathbf{x})]^2 \ge \tau, \tag{6.44}$$

where $\tau \ge 0$ is a threshold, typically of the order of the square of radius R_K. It is interesting to remark that, while in the case of the linear kernel and polynomial kernels all the

terms in the l.h.s. of (7.11) need to be computed for each point, for the Hausdorff kernel and in general for all kernels so that $K(\mathbf{x}, \mathbf{x})$ is constant, the only term which depends on point $\mathbf{x}$ is the second, the other two being constant. In this case, $\mathbf{x}$ is detected as a novelty if the inequality

$$\sum_{j=1}^{\ell} \alpha_j K(\mathbf{x}, \mathbf{x}_j) \geq \tau \tag{6.45}$$

is violated, for some fixed value of τ.

6.4.1 Another kernel-based method for novelty detection

An alternative approach to novelty detection has been developed by Schölkopf et al. in [SPST+00].

The method is based on using a kernel function with the property that $\langle \boldsymbol{\phi}(\mathbf{x}) \cdot \boldsymbol{\phi}(\mathbf{x}) \rangle = K(\mathbf{x}, \mathbf{x}) = c$. In this special case, in feature space the data lie in a region on the surface of a hyperplane. The objective is therefore to separate off this region from the surface region containing no data.

This is achieved by constructing a hyperplane $\mathbf{w} \cdot \mathbf{x}_i - \rho = 0$ which is maximally distant from the origin with all data $\mathbf{x}_i$ lying on the opposite side of it from the origin, such that $\mathbf{w} \cdot \mathbf{x}_i - \rho \geq 0$. As usual, to handle noise and outliers we introduce slack variables, ξ_i. This leads to the following QP problem:

$$\textbf{Problem ND2} \qquad \begin{array}{ll} \min_{w,\rho,\xi_i} & \frac{1}{2}||\mathbf{w}||^2 - \rho + C\sum_{i=1}^{\ell} \xi_i \\ \text{subject to} & (\mathbf{w} \cdot \phi(\mathbf{x}_i)) \geq \rho - \xi_i \\ & \xi_i \geq 0. \end{array} \tag{6.46}$$

Using the multipliers $\alpha_i, \beta_i \geq 0$, we again introduce the primal Lagrangian

$$L(\mathbf{w}, \rho, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\xi}) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{\ell} \xi_i - \rho \sum_{i=1}^{\ell} \alpha_i (\mathbf{w} \cdot \phi(\mathbf{x}_i) - \rho + \xi_i) - \sum_{i=1}^{\ell} \beta_i \xi_i$$

which yields

$$\sum_{i=1}^{\ell} \alpha_i = 1, \tag{6.47}$$

$$\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i \phi(\mathbf{x}_i), \tag{6.48}$$

$$\beta_i = C - \alpha_i, \qquad \forall i = 1, \ldots, \ell \tag{6.49}$$

The dual problem becomes

$$\begin{array}{lll} \textbf{Problem DND2} & \min_{\alpha} & \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \\ & \text{subject to} & \sum_{i=1}^{\ell} \alpha_i = 1 \\ & & 0 \leq \alpha_i \leq C. \end{array} \tag{6.50}$$

ρ can be recovered from the Kuhn-Tucker conditions: for some $\mathbf{x}$ so that the multiplier associated α is $0 < \alpha < C$,

$$\rho = \mathbf{w} \cdot \phi(\mathbf{x}) = \sum_{i=1}^{\ell} \alpha_i K(x_i, x). \tag{6.51}$$

6.4.2 Comparison between the two methods

It can be shown that, for a kernel K with the following property

$$K(\mathbf{x}, \mathbf{x}) = \text{constant}, \quad \forall \ \mathbf{x} \ \text{ of the input space}$$

Problem **ND1** and Problem **ND2** are equivalent. To prove this we need to show that both the functions to minimize and the constraints are equivalent.

Equivalence between the functions

We derive R^2 from Equation (6.43), and then insert it in the minimization function of Problem **ND1**, where $\mathbf{x}$ is a vector of the training set so that $0 < \alpha < C$:

$$R^2 + C\sum_{i=1}^{\ell}\xi_i = K(\mathbf{x},\mathbf{x}) + \sum_{j,k=1}^{\ell}\alpha_j\alpha_k K(\mathbf{x}_j,\mathbf{x}_k) - 2\sum_{j=1}^{\ell}\alpha_j K(\mathbf{x},\mathbf{x}_j) + C\sum_{i=1}^{\ell}\xi_i \qquad (6.52)$$

If $K(\mathbf{x},\mathbf{x})$ is constant, through (6.51), Equation (6.52) is equivalent to

$$\sum_{j,k=1}^{\ell}\alpha_j\alpha_k K(\mathbf{x}_j,\mathbf{x}_k) - 2\sum_{j=1}^{\ell}\alpha_j K(\mathbf{x},\mathbf{x}_j) + C\sum_{i=1}^{\ell}\xi_i = ||\mathbf{w}||^2 - 2\rho + C\sum_{i=1}^{\ell}\xi_i$$

which is equivalent to the minimization of Problem **ND2**.

Equivalence between the constraints

Starting from the constrains of Problem **ND1**, in feature space ($i = 1,\ldots,\ell$):

$$K(\mathbf{x}_i,\mathbf{x}_i) + \sum_{j,k=1}^{\ell}\alpha_j\alpha_k K(\mathbf{x}_j,\mathbf{x}_k) - 2\sum_{j=1}^{\ell}\alpha_j K(\mathbf{x}_i,\mathbf{x}_j) \leq R^2 + \xi_i.$$

R^2 is derived from Equation (6.43), where $\mathbf{x}$ is a vector of the training set so that $0 < \alpha < C$:

$$K(\mathbf{x}_i,\mathbf{x}_i) + \sum_{j,k=1}^{\ell}\alpha_j\alpha_k K(\mathbf{x}_j,\mathbf{x}_k) - 2\sum_{j=1}^{\ell}\alpha_j K(\mathbf{x}_i,\mathbf{x}_j) \leq K(\mathbf{x},\mathbf{x}) + \sum_{j,k=1}^{\ell}\alpha_j\alpha_k K(\mathbf{x}_j,\mathbf{x}_k) - 2\sum_{j=1}^{\ell}\alpha_j K(\mathbf{x},\mathbf{x}_j) + \xi_i$$

which, in case the hypothesis of constancy of $K(\mathbf{x}, \mathbf{x})$ holds, becomes:

$$2\sum_{j=1}^{\ell} \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 2\sum_{j=1}^{\ell} \alpha_j K(\mathbf{x}, \mathbf{x}_j) + \xi_i$$

which is equivalent to the constraints of Problem **ND2**.

6.5 Kernel requirements

Let us now focus on the QP Problems **DCL2** and **DND1**, and consider under which conditions they have a unique solution. Typically, the positive defineteness of the function K selected is required; if this holds true — the function K is a Mercer's kernel — the objective functions of Problems **DCL2** and **DND1** are convex whichever are the parameters α_i selected. This raises the question of whether it is possible to find weaker conditions on K such that inequality (6.3) is satisfied subject to certain conditions on the coefficients α_i. Indeed, a closer look to the QP Problems **DCL2** and **DND1** reveals that the uniqueness of the solution would be ensured by the convexity of the objective function in the feasible region. In the case of binary classification, the uniqueness of the solution is ensured by the following (ignoring a common scaling factor):

$$\begin{aligned} &\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0, \\ &\text{subject to } \sum_{i=1}^{\ell} \alpha_i = 0 \end{aligned} \tag{6.53}$$

Instead, in the case of novelty detection, we have:

$$\begin{aligned} &\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0, \\ &\text{subject to } \alpha_i \geq 0, \;\; i = 1, \ldots, \ell. \end{aligned} \tag{6.54}$$

In particular, in the case of binary classification, conditionally positive definiteness of a function K is sufficient for it to be a positive definite function on the feasibile region of the QP Problem **DCL2**. We remind that a function K is *conditionally positive definite* of order 1 if, for each choice of $\mathbf{x}_1, \ldots, \mathbf{x}_n$, and each choice of α_i, $i = 1, \ldots, n$, so that $\sum_{i=1}^{n} \alpha_i = 0$, $\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0$.

In the case of novelty detection the positivity of the function K is a sufficient condition for K to be positive definite, considering the positivity constraints of the QP Problem **DND1** on the coefficients α_i.

Chapter 7

A kernel for grey-level images

Learning one class at a time can be seen as an effective solution to classification problems in which only the positive examples are easily identifiable. In this chapter we show that the design of an appropriate image similarity measure makes it possible to solve a multiclass classification problem effectively even in the absence of negative examples, by means of the kernel method for learning one class at a time described in Chapter 6. To this purpose, we introduce a function which is based on the similarity method described in Chapter 5, tailored to deal with grey-level image matching. We prove that this function can be used as a kernel for novelty detection, although it is not a Mercer's kernel. We demonstrate its effectiveness on several data sets of faces and of 3D objects of artistic relevance, like statues. We finally show how, by carefully controlling the amount of dilation, this function can be modified in a function which is a Mercer's kernel.

7.1 Introduction

In the *learning from examples* paradigm, the goal of many detection and classification problems of computer vision — like object detection and recognition (see [PP00, PV98, RJV99, RBK98, SK00, WFKvdM97] for example) — is to distinguish between positive and negative examples. While positive examples are usually defined as images or portion of images containing the object of interest, negative examples are comparatively far less expensive to collect but somewhat ill defined and difficult to characterize as a class. For this

reason, a representative list of informative negative examples is usually obtained by carefully selecting the most difficult negative examples or most likely false positives [PP00].
If the examples belong to only one class, the idea is that of determining the spatial support of the available data by finding the smallest sphere in feature space enclosing the positive examples [CB01]. The feature mapping, or the choice of the kernel, plays a crucial role here.
The topic of kernels for images is relatively new. A number of studies have been reported about the use of general purpose kernels for image-based classification problems [PV98, PP00, GLK00, JMKL00, HSPP01, MPP01], while a family of functions which seem to be better suited than Gaussian kernels for dealing with image histograms has been studied in [CHV99].
In this chapter we introduce a kernel derived from the image matching technique described in Chapter 5, well suited to capturing image similarities while preserving meaningful image differences. We present experiments which show that this kernel outperforms the linear kernel (effectively corresponding to a standard template matching technique), polynomial kernels of various degrees and Gaussian RBF kernels in the representation and the identification of 3D objects. This is consistent with the observation that the general purpose kernels are all based on a pointwise match between input points, while the images we used are not in pointwise correspondence. The efficacy of the method is assessed on databases of faces and 3D objects of artistic interest. Not surprisingly, the best results for the Hausdorff kernel are obtained allowing for some dilation along the spatial dimensions, dilation which compensates for image misalignment and small grey level changes across the sequence.
In summary, the major aim of this chapter is to evaluate the appropriateness of Hausdorff-like measures for engineering kernels well adapted to deal with image related problems. A secondary objective is to assess the potential of kernel methods for novelty detection in computer vision problems (which are problems in high dimensional spaces with a relatively small numbers of positive examples).

7.2 Kernels for images

For years, most of the research in machine learning for images has been based on preliminary feature extraction: images are preprocessed, and their content represented in more compact ways, for instance, by means of edges, features, deformable models, or higher level shape descriptions. This is mainly due to the fact that object recognition systems use standard distance metrics like weighted norms or correlation, therefore the main factor which distinguishes different approaches is the input representation. Also, traditional classification approaches perform poorly when working directly on images because of the high-dimensionality of the data, but SVMs and other kernel methods have been shown to be useful for these applications, since they can avoid the difficulties caused by very high-dimensional representations. This has been illustrated in publications including among others, [CV95, SBV95] for hand-written digit recognition, [Joa98] for the case of text categorization, and [PV98] for image-based object recognition.

Even when working directly on images, usually some preprocessing is used: images are subsampled, filtered or registered. Often sub-images containing relevant features are extracted from the original images: in face recognition, for instance, several image based methods use images of eyes, nose, mouth more than images of the full face.

In our approach preprocessing efforts are reduced. We explore the possibility of representing objects by means of images, which we simply rescale to meet complexity issues. The idea behind this choice is to exploit fully the potential of kernel methods to transfer the intrinsic complexity of a problem into the design of a kernel.

We believe that the knowledge inherited from decades of work in image processing and computer vision can be transferred into the study and the design of kernels for images, which can ease classification tasks performed on input data which are images.

As regards data representation, an $N \times M$ image I will be represented simply as a vector $\mathbf{x}$ of a space of dimension NM, where each row of the image will be put side by side with its preceeding and its succeeding row.

7.2.1 Linear kernel and correlation methods

If we restrict the input data space to vectors generated by a specific phenomenon (image acquisition, in our case), then we can attempt to give a more intuitive interpretation of the kernel functions used, in terms of the input data. If the input data are images, a kernel, which takes two images and returns a value that is the inner product in the feature space, can be seen as a similarity function. It still remains to establish whether the similarity functions used by the computer vision community, are kernels. Again, as in Section 5.2, we do not consider measures based on geometric transformations or local ordering information, instead we restrict our attention on area-based similarity measures which can be directly applied to grey-level patches. The correlation methods introduced in Section 5.2 are all potential kernels for images. For example, if images are represented as vectors of a NM-dimensional space, cross-correlation, described by the score function of Equation (5.2), is an inner product between two input vectors, and thus a kernel — specifically, the linear kernel of Equation (6.4). Normalized cross-correlations are also kernels, since they derive from cross-correlation as in the property described by Equation (6.9).

It is not straightforward, however, to demonstrate that the Sum of Squared Differences, described by Equation (5.3), is a kernel. We first write Equation (5.3) in the following way ($\mathbf{x}_1, \mathbf{x}_2 \in X \subseteq \mathbb{R}^{NM}$):

$$\begin{aligned} K_{ssd} &= -||\mathbf{x}_1 - \mathbf{x}_2||^2 = \\ &= -\sum_{i=1}^{NM} ((x_1)_i - (x_2)_i)^2 \end{aligned} \tag{7.1}$$

It is obvious that the above function is not positive definite. We demonstrate that it is conditionally positive definite of order 1, that is, that for each choice of $\mathbf{x}_1, \ldots, \mathbf{x}_n$, it satisfies inequality (6.3) under the constraint

$$\sum_{i=1}^{n} \alpha_i = 0. \tag{7.2}$$

Indeed, if (7.2) holds true,

$$\begin{aligned}
&\sum_{i,j=1}^{n} \alpha_i \alpha_j K_{ssd}(\mathbf{x}_i, \mathbf{x}_j) = \\
= \; & -\sum_{i,j=1}^{n} \alpha_i \alpha_j ||\mathbf{x}_i - \mathbf{x}_j||^2 = \\
= \; & -\sum_{i=1}^{n} \alpha_i \sum_{k=1}^{MN} (\mathbf{x}_i)_k^2 \sum_{j=1}^{n} \alpha_j - \sum_{j=1}^{n} \alpha_j \sum_{k=1}^{MN} (\mathbf{x}_j)_k^2 \sum_{i=1}^{n} \alpha_i + 2 \sum_{i,j=1}^{n} \alpha_i \alpha_j \sum_{k=1}^{MN} (\mathbf{x}_i)_k (\mathbf{x}_j)_k = \\
= \; & 2 \sum_{i,j=1}^{n} \alpha_i \alpha_j \langle \mathbf{x}_i \cdot \mathbf{x}_j \rangle.
\end{aligned}$$

If α_i sum to 0, proving that K_{ssd} is a positive definite function is reduced to prove that the inner product is positive definite. As anticipated in Section 6.5, a conditionally positive definite function can be used as a legitimate kernel for binary classification without being a Mercer's kernel, because it always leads to a semi-definite positive Gram matrix if the coefficients α_i are chosen in the feasible region of Problem **DCL2**.

7.2.2 Hausdorff kernels

In this section we study the conditions under which the Hausdorff-based similarity method introduced in Chapter 5 (Equation (5.5)) is a legitimate kernel.
Using the current data representation the equation can be rewritten in the following way ($\mathbf{x}_1, \mathbf{x}_2 \in X \subseteq \mathbb{R}^{NM}$):

$$H(\mathbf{x}_1, \mathbf{x}_2) = \sum_{i=1}^{MN} \theta(\epsilon - \min_{(\mathbf{x}_2)_j \in N_{(\mathbf{x}_1)_i}} |(\mathbf{x}_1)_i - (\mathbf{x}_2)_j|). \tag{7.3}$$

As we remarked in Section 5.3, the function described by Equation (5.5) is not symmetric, but symmetry can be restored, for instance, in the following way:

$$K_H \;\; = \;\; \frac{1}{2} \left[\{ H(\mathbf{x}_1, \mathbf{x}_2), H(\mathbf{x}_2, \mathbf{x}_1) \} \right]. \tag{7.4}$$

Let us now consider more explicitly the case where the data sets are images. Similarly to

Chapter 5 we can consider two grey-level images A_{im} and B_{im} and sketch a computation scheme:

1. Expand the two images A_{im} and B_{im} into 3-D binary matrices A and B respectively, the third dimension being the grey value. For example, for A we write

$$A[i,j,k] = \begin{cases} 1 & \text{if } A_{im}[i,j] = k; \\ 0 & \text{otherwise.} \end{cases} \tag{7.5}$$

   and similarly for B. Notice that a 3-D matrix like $A[i,j,k]$ for each choice of (i,j) equals 1 only for one value of k.

2. Dilate both matrices by growing their nonzero entries by a fixed amount $\epsilon/2$ in the grey value dimension and half the linear size of the neighborhood N in the spatial dimensions. Let $\tilde{A}$ and $\tilde{B}$ the resulting 3-D dilated binary matrices. This dilation varies according to the degrees of similarity required and the transformations allowed.

3. To obtain $K_H(A_{im}, B_{im})$ compute the size of the intersections between A and $\tilde{B}$, and B and $\tilde{A}$, and take the average of the two values obtaining $K_{Hau}(A_{im}, B_{im})$. Thinking of the 3-D binary matrices as (binary) vectors in obvious notation we could also write

$$K_H(A_{im}, B_{im}) = \frac{1}{2}\left(\mathbf{A}\cdot\tilde{\mathbf{B}} + \mathbf{B}\cdot\tilde{\mathbf{A}}\right) \tag{7.6}$$

We immediately notice that, by construction, $K_H(\mathbf{x}_i, \mathbf{x}_j) \geq 0$, for each $\mathbf{x}_i$, $\mathbf{x}_j$; therefore this is one of the cases discussed in Section 6.5, and the constraints (6.54) are satisfied for all α_i of the feasible region. Consequently, the Hausdorff similarity measure K_H can *always* be used as a kernel for novelty detection, even if *it is not* a Mercer's kernel. Experimental results obtained with K_H in the case of novelty detection will be shown in Section 7.3.
In the case data are images, the fact that K_H is not a Mercer's kernel can be seen through a simple counterexample; let A_1, A_2, and A_3, be 3 1-pixel images

$$A_1 = 10; \quad A_2 = 12; \quad A_3 = 14.$$

If dilation is $\epsilon = 3$, then

$$K_H(A_1, A_2) = 1; \quad K_H(A_1, A_3) = 0, \quad K_H(A_2, A_3) = 1,$$

and the kernel matrix $\mathbf{K}_{ij} = K_H(A_i, A_j)$ for $i, j = 1, \ldots, 3$

$$\begin{matrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{matrix}$$

has a negative determinant.

Let us now show that by controlling the amount of dilation it is possible to obtain a modified similarity measure which is a Mercer's kernel. Neglecting "boundary entries" and denoting with n the number of entries in each neighbourhood, the key is to *redefine* the dilation step as

$$\tilde{A}'[i, j, k] = A[i, j, k] + 1/n \sum_{i', j', k'} A[i', j', k'] \geq A[i, j, k] \tag{7.7}$$

where the sum ranges through the entries the neighborhoods of which contain the entry (i, j, k). Note that if $\tilde{A}'[i, j, k] \neq 0$ then $\tilde{A}'[i, j, k] \geq 1/n$. Unlike the previous definition, the new dilation is obtained combining linearly various contributions with a weight strictly less than 1.

If we now define

$$H'(A_{im}, B_{im}) = \mathbf{A} \cdot \tilde{\mathbf{B}}'$$

it can be shown that the similarity measure

$$K'_H(A_{im}, B_{im}) = \frac{1}{2} \left(H'(A_{im}, B_{im}) + H'(B_{im}, A_{im}) \right) = \frac{1}{2} \left(\mathbf{A} \cdot \tilde{\mathbf{B}}' + \mathbf{B} \cdot \tilde{\mathbf{A}}' \right) \tag{7.8}$$

is an inner product and thus a Mercer's kernel.

In the next section the interested reader may find the proof of the fact that K'_H is a Mercer's kernel along with a discussion of the differences and similarities between expressions

(7.6) and (7.8).

7.2.3 Discussion

Here we prove that the function K'_H of equation (7.8) is a Mercer's kernel by showing that it is an inner product.

We immediately see that symmetry and linearity follow easily from the fact that K'_H is computed using standard inner products. For the nonnegativity we need to show that

$$\mathbf{M} \cdot \tilde{\mathbf{M}}' \geq 0 \tag{7.9}$$

for an *arbitrary* matrix M (not necessarily binary or corresponding to an image) and its dilation $\tilde{M}'$ obtained as in (7.7).

If we denote with M_p the p-th component of the P-dimensional vector $\mathbf{M}$ (in some 1-to-1 correspondence with the entry (i, j, k) of M), plugging (7.7) in (7.9) with $\mathbf{A} = \mathbf{M}$ gives

$$\frac{1}{2n}\left(2n\sum_{p=1}^{P} M_p^2 + 2\sum_{p=1}^{P}\sum_{i=1}^{n} M_p M_{q_{p,i}}\right)$$

where the components $M_{q_{p,1}}, ..., M_{q_{p,n}}$ identify the n components of $\mathbf{M}$ corresponding to the entries (i', j', k') in (7.7). Neglecting the $(1/2n)$ factor, if we expand and rearrange the terms of the above sums we obtain

$$\underbrace{M_1^2 + ... + M_1^2}_{2n} + \underbrace{M_2^2 + ... + M_2^2}_{2n} + ... + \underbrace{M_P^2 + ... + M_P^2}_{2n} + 2M_1\left(M_{q_{1,1}} + ... + M_{q_{1,n}}\right) +$$

$$2M_2\left(M_{q_{2,1}} + ... + M_{q_{2,n}}\right) + ... + 2M_P\left(M_{q_{P,1}} + ... + M_{q_{P,n}}\right).$$

Since for each $p = 1, .., P$ there are n mixed products of the type $M_p M_{q_{p,i}}$ $i = 1, ..., n$ and n of the type $M_{q_{p,i}} M_p$, is easy to conclude that the whole expression can be written as a sum of $P \times n$ squares, with each square term of the type

$$(M_p + M_q)^2$$

thereby proving that inequality (7.9) is always true.

Two comments are in order. First, it is clear that the number and relative weight of the various terms make it possible to obtain a sum of squares because the dilation contributions originating from the same entry sum up to 1. This is actually the only constraint, as different and space variant dilation contributions (provided their sum does not exceed 1) would not change the substance of the proof. Notice that if the sum exceeds 1 the nonnegativity property is lost. As a second final remark we observe that for relatively large images and dilation neighborhood of small size, even the function K_H, though not a positive definite function, in practice often leads to kernel matrices which are positive semidefinite.

We conclude this section discussing the relation between the Mercer's kernel of (7.8) and the Hausdorff similarity measure defined by (7.4). First we denote with $\delta(i,j)$ the Kronecker $\delta_{A_{im}[i,j]B_{im}[i,j]}$, which equals 1 if $A_{im}[i,j] = B_{im}[i,j]$ and 0 otherwise. Then, using the same notation of before for $H'(A_{im}, B_{im})$ we have

$$\sum_{i,j=1}^{N} w_{ij} \left(\delta(i,j) + (1 - \delta(i,j)) U(\epsilon - |A_{im}[i,j] - B_{im}[\hat{i},\hat{j}]|) \right) \tag{7.10}$$

with

$$w_{ij} = \tilde{B}'[i,j,A_{im}[i,j]].$$

Similarly to (7.4), also in this case expression (7.10) counts the number of pixels (i,j) of A_{im} the grey value of which differ by no more than $\epsilon > 0$ from at least one grey value of B_{im} over the neighborhood O. Two are the main differences between expressions (7.4) and (7.10). First, consistently with the new definition of dilation, in (7.10) the case in which the two images takes on exactly the same value at the same location is treated separately. The second difference is that to each match is also attributed a weight proportional to the value of the dilation. The weight w_{ij} for exact matches is never less than 1 because if $A_{im}[i,j] = B_{im}[i,j]$ using (7.7) and (7.5) with $A = B$ we have

$$w_{ij} \geq B[i,j,A_{im}[i,j]] = 1.$$

For nonexact matches, instead, $B[i,j,A_{im}[i,j]] = 0$ but $\tilde{B}'$ (or w_{ij}) is at least $1/n$. The Kronecker δ ensures that for each pair (i,j) only one of the two unit step functions is strictly positive.

7.3 Experiments

In applications based on describing an object or a scene by means of a collection of images, the training set might consist of image sequences in which the spatio-temporal intensity differences between close frames are rather small. At run time, the problem could be to classify a new image as a novel view of the same scene possibly partly occluded or slightly changed due to a number of different reasons. As proposed in [BOV02], one way to tackle this problem is through the use of novelty detection. A relatively large number of views of the same object or scene are gathered and used to build the sphere in feature space enclosing the examples. In this section we present results to this approach obtained on several image sequences: data sets of faces for face recognition, and data sets of 3D objects of artistic interest acquired in San Lorenzo Cathedral. The two problems are closely related but distinct. As we will see in the case of the latter, the notion of object is blurred with the notion of scene (the background being as important as the foreground), while in face recognition this is not the case.

In our approach we make a minimal use of preprocessing. In particular, we do not compute accurate registrations, since our similarity measure takes care of spatial misalignments. Also, since our mapping in feature space allows for some degree of deformation both in the grey-levels and in space, the effects of small illumination and pose changes are attenuated. Finally, we exploit full 3D information on the object by acquiring a training set which includes frontal and lateral views.

In the remainder of this section we will first describe the recognition system, and then the results of its application on the two different families of data sets. All the images used have been resized to 72×58 pixels, we are therefore working in a space of more than 4000 dimensions.

7.3.1 The recognition system

In the case of novelty detection, the similarity function K_H for grey level images of Equation (7.4) can be used as a kernel. Given a training set of images of the object of interest, we estimate the smallest sphere containing the training data in the feature space implicitly defined by the kernel for images, K_H. After training, a new image $\mathbf{x}$ is classified as a positive example if

$$K(\mathbf{x},\mathbf{x}) - 2\sum_{j=1}^{\ell}\alpha_j K(\mathbf{x},\mathbf{x}_j) + \sum_{j,k=1}^{\ell}\alpha_j\alpha_k K(\mathbf{x}_j,\mathbf{x}_k) = [d_K(\mathbf{x})]^2 \leq \tau, \tag{7.11}$$

where $\tau \geq 0$ is a threshold, typically of the order of the square of radius R_K. It is interesting to remark that, while in the case of the linear kernel and polynomial kernels all the terms in the l.h.s. of (7.11) need to be computed for each point, for the Hausdorff kernel the only term which depends on point $\mathbf{x}$ is the second, the other two being constant. In this case, $\mathbf{x}$ is detected as a novelty if the inequality

$$\sum_{j=1}^{\ell}\alpha_j K(\mathbf{x},\mathbf{x}_j) \geq \tau \tag{7.12}$$

is violated.

In the experiments shown in this section, the performances of K_H are compared with the performances of standard kernels, as the linear kernel (Equation (6.4)) and the polynomial kernel (Equation (6.6) with $d = 2, 3$). After training the system with positive examples only, the performances at different τs are evaluated on a test set of both positive and negative examples, and described by means of Receiver Operating Characteristic (ROC) curves [GS74]. Each point of a ROC curve represents a pair consisting of the false-alarm rate and the hit-rate of the system, for some value of the threshold τ of Equation (7.11). The system efficiency can be evaluated by the growth rate of its ROC curve, and for a given false-alarm rate, the better system will be the one with the higher hit probability. The overall performance of a system can be measured by the area under the curve. The optimal τ for a certain recognition task is usually chosen as the $\hat{\tau}$ corresponding to the

so called equal error rate (E.E.R.), obtained when the percentage of false positives rate equals the percentage of false negatives rate (miss rate).

7.3.2 Is there any need for image preprocessing?

One of the major aims of our object representation is to keep preprocessing minimal. Although, it is known that in some cases foreground segmentation cannot be avoided; accurate segmentation can be a time consuming task, when no a priori actions could make the operation easier: for instance, in cultural heritage applications, there is no way of intervening on the scene, with special lighting or artificial background for instance, in order to facilitate preprocessing. Often segmentation is performed manually, whenever the appearance of the scene does not allow to use reliably common segmentation techniques, neither based on color or texture distribution, nor based on edges. The fourth dimension, time, can be of some help, but it is not a general solution. If the 3D dynamics of the scene is strong, a 3D structure or 3D motion approximation can be used to produce some layered representation of the scene, which brings to a coarse segmentation. Section 3.3 describes a technique which we experimented as a possible automatic solution to object segmentation. As pointed our in Section 3.3, the approach we chose, lacks generality, and more robust layered representations would be needed [Ade91], to deal with more complicated scene structures. As an alternative, we investigated the possibility of abandoning an automatic approach, in favour of a more general solution. Our final choice went for a semi-automatic solution, that exploits the spatio-temporal contiguity of image sequences. We reasoned on the fact that, if at a higher level we will use a similarity measure which is tolerant to occlusions, the result of segmentation does not have to be too accurate. In practice, more than a segmentation we produce an image reduction, that is we approximate each 2D shape projection of $\mathcal{O}$ with a rectangular shape inscribing the shape (or an interesting part of it). The procedure we apply is semi-automatic:

1. We manually select a rectangular patch from the first image I_1 of the sequence

2. We use it as a model M_1 to search inside all the subsequent images of the sequence.

The window M_k of each image I_k that represents the best match of M_1 is cut and saved as a reduced version of I_k.

3. When the match value gets lower than a fixed threshold T, then the view I_i is not adequately represented by the model M_1. It is the time to select manually a new model from image I_i and restart the process from step 2., on the remaining part of the sequence.

Image frames which undergo manual selection represent changes with respect to the previous frames. The 3D path of the camera is improvised and thus not repetible. In general we do not make any assumption on instrinsic parameters, for instance we do not ask for the focal length to be fixed.

This simple method produces a new sequence (the sequence made of all the reduced images M_k, $K = 1, 2, \ldots$) that contains only useful information. Obviously this selection of views is not unique, because the choice of the starting point is random and the result depends on a number of thresholds (both the thresholds of the similarity measure and the threshold used to decide when the model has to be changed.

The procedure described above has been used for the face identification experiments described in Section 7.3.4. In this case it has mainly used to cut the sequence automatically, since in some cases the background was occupying a great part of the images. If we extend our face identification system to a less constrained environment (for instance by allowing background variations), the same preprocessing would be useful to cut down the effect of changes in the background.

There are some cases, though, when we wonder if such preprocessing is necessary. Consider the images shown in Figure 7.1, belonging to one of the training sets of statues acquired in San Lorenzo Cathedral; note the following facts:

- The texture of the foreground and the background is too similar to consider static segmentation techniques;

- The 3D dynamic of the scene is too poor to exploit the effect of parallax, with motion-based segmentation;

Figure 7.1: Samples from a training set representing a rigid 3D object (a statue) captured in its natural environment (a church). This example shows how, in the case the objects described are left permanently in a certain environment, the background itself can bring additional useful information to the recognition task.

- The environment where the objects are located (a church) does not allow for any modification, temporary or permanent, neither of the illumination nor of the background.

Nevertheless, the objects in question have been located in the same position for decades or even centuries. This consideration make us think that maybe it is not necessary to perform any segmentation at all. The idea of object can be blurred with the idea of scene, and the background, instead of being an obstacle to the recognition task becomes another vehicle of information. Before we decide to skip segmentation we should always consider two important details: the first one is the invariance of the scene to major transformations, the second one is the 3D dyamics of the scene – if the distance between the object and the background is too big, a small pose variation of the camera will produce wide occlusions.

7.3.3 Application to 3D object recognition

In this application we aim to represent and identify 3D objects against a complex background. While the representation is based on an image sequence containing differing views of the object of interest, the recognition consists of finding object occurences in novel single views. For this experiment, we acquired video sequences of 3D objects in San Lorenzo Cathedral. The objects are marble statues, and, in particular, the ones we

used for these experiments are all located in the same chapel (Cappella di S. Giovanni Battista) and thus acquired under similar illumination conditions: this results in noticeable similarities between the brightness patterns of all the images. As discussed in Section 7.3.2, under these circumstances segmentation is inadvisable, since the background itself is characterizing of the object. In the case of San Lorenzo Cathedral, we can safely assume that the statues will not be moved from their usual position.

Figure 7.3 shows samples of the training sets used to capture the 3D appearance of two of the six statues; in what follows we will refer to these two statues as statue A and statue B. Figure 7.4 illustrates samples of the negative examples with respect to both statues A and B. Notice that some of the negative examples look similar to the positive ones, even to a human observer, especially at the image sizes used in the representation or training stage (58×72).

Kernels evaluation

Table (7.1) presents the equal error rates – or the error rates obtained for the value of the threshold τ for which the percentages of false negative and false positive are the same – for several kernels on one of the training sets. By inspection, it can clearly be seen that the Hausdorff kernel outperforms the best results obtained with general purpose kernels. This is consistent with the observation that the general purpose kernels are all based on a pointwise match between input points, while the images we used are not in pointwise correspondence. Not surprisingly, the best results for the Hausdorff kernel are obtained allowing for some dilation along the spatial dimensions, dilation which compensates for image misalignment and small grey level changes across the sequence.

The last two columns of Table (7.1) show the maximum and minimum values of each kernel matrix on the training set for statue 4. The values of the linear kernel and of the sum of square differences kernel of Equation (5.3) can be used to appreciate the difficulty of the task and the explain the reason of the large gap between the behavior of the Hausdorff kernel and the general purpose kernels. The difference between the maximum and minimum values for both kernels tells us that all the training images are strongly correlated.

kernel	***e.e.r.*** (%)	$K_{\max}$	$K_{\min}$
Hausdorff Δ = (0,0,0)	19	4.176	0.010
Hausdorff Δ =(1,0,0)	15	4.176	0.033
Hausdorff Δ =(3,0,0)	11	4.176	0.088
Hausdorff Δ =(1,1,1)	4	4.176	0.257
Hausdorff Δ =(3,1,1)	3	4.176	0.457
Hausdorff Δ =(4,3,3)	4	4.176	1.489
Sum of sq. diff.	48	0.000	-5231
linear	36	0.734	0.320
2-nd deg polynomial	37	2.009	0.743
3-rd deg polynomial	39	4.219	1.300
Gaussian ($\sigma = 1000$)	50	1.000	0.000
Gaussian ($\sigma = 3000$)	28	1.000	0.055
Gaussian ($\sigma = 5000$)	52	1.000	0.351
Gaussian ($\sigma = 10000$)	47	1.000	0.770

Table 7.1: Equal error rates (*e.e.r.*) for SVMs with different kernel on the training set of statue A. For the Hausdorff kernel, different values for the size of the dilation Δ along the grey level and spatial dimensions respectively are shown, and the matrix entries were scaled by a factor $n = 10^{-4}$. For the linear and polynomial kernels, the inputs were scaled by a factor $n = 10^{-4}$. For the Gaussian kernel, we tried different σs. The last two rows report the maximum and minimum value of each matrix in the training set.

Since each image consists of 58×72 pixels, from the minimum value of the sum of square difference kernel matrix, for example, we gather that the average grey value difference between the two most different training images is less than 2 grey values.

Figure 7.2 shows a grey level representation of some of the kernel matrices obtained using the kernels of Table 7.1. Notice the strong diagonally dominant structure of the matrices induced by both the Hausdorff kernel with no dilation and the Gaussian kernel for smaller σs. The effect is definitely attenuated by allowing for some dilation along the grey level and (especially) along the spatial dimension for the Hausdorff kernel and for larger σ for the Gaussian kernel. Notice however the different impact of this procedure on the recognition rates. Clearly, while in the Hausdorff case the matrix structure reflects the fact that close frames are similar, in the Gaussian case larger σs lead to kernel matrices with poor discriminability.

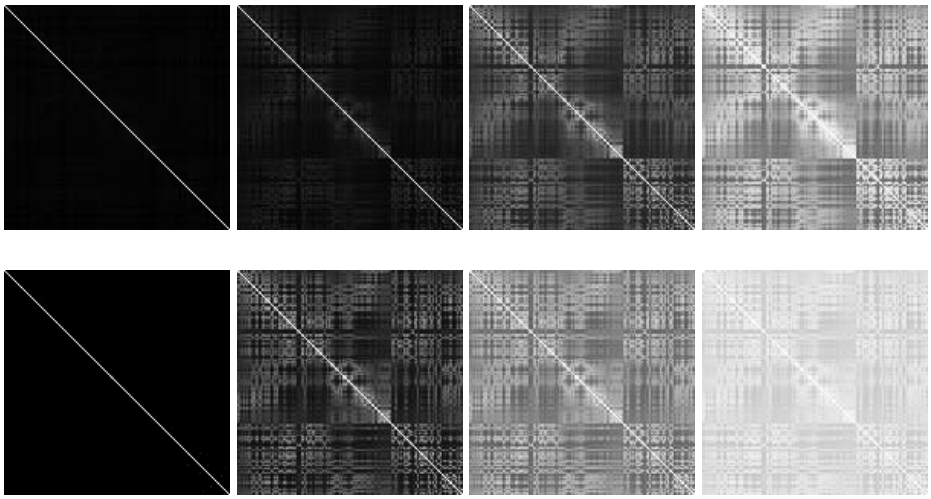


Figure 7.2: Grey value representation of the kernel matrices of the Hausdorff kernel for dilation (0,0,0), (3,0,0), (3,1,1), and (4,3,3) (top row) and of the Gaussian kernel for i $\sigma = 1000, 3000, 5000$, and 10000 (bottom row). Larger values correspond to lighter grey levels.

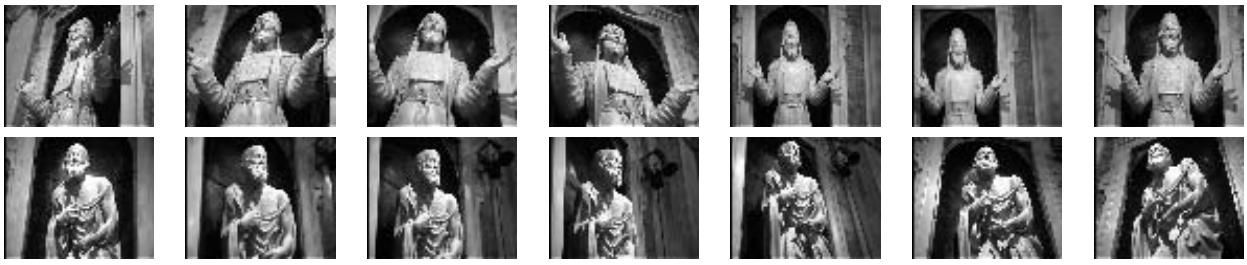


Figure 7.3: Samples from two training sets for 2 different statues.

Recognition performances

Figures 7.5 and 7.6 compare the recognition performances for the most promising kernels (see Table (7.1): Hausdorff kernel ($\Delta = (4,3,3)$), polynomial ($d = 1,2,3$), Gaussian RBF ($\sigma = 3000$). Notice that the results of our kernel are still very good, even with the smaller training sets.

To evaluate the system from the point of view of its representation ability, we trained it with training sets increasingly small to check for graceful degradation of the results. The

Figure 7.4: A few negative examples, with respect to both statues of Figure 7.3.

curves on the top rows have been obtained with larger training sets (314 images for statue A and 226 images for statue B) than the ones on the bottom rows (97 images for statue A and 95 images for statue B).

Moreover, to test the system from the point of view of its classification potential, we also searched for the six statues within various images acquired from differing view points and distances. We performed a multiscale search for all of the six statues, in three steps. For each one of the six classifying systems $\mathcal{C}_i$, $i = 1, \ldots, 6$:

1. We consider the test images at n different scales ($n = 30$ in our experiments);
2. We shift a fixed size window (58×72) across all the rescaled version of each test image.
3. We then classify the content of the window with $\mathcal{C}_i$.

Figure 7.7 shows the results on a few frames from a panning sequence of the identification of statue A and B. The appropriately resized white and grey rectangles are the matches of statue A and B, respectively.

7.3.4 Application to face identification

In this application we assume a collaborative scenario in which the subjects aim at being recognized by the system, which can therefore always view them from approximately the same view point and distance. We may therefore employ training and test images of the same fixed size.

First data set

In the first set of images, we acquired both training and test data in the same session. We collected four sets of images (frontal and rotated views), one for each of four subjects, for a total of 353 images, samples of which are shown in Figure 7.8. To test the system we used 188 images of ten different subjects, including test images of the four people used to acquire the training images (see the examples in Figure 7.9).

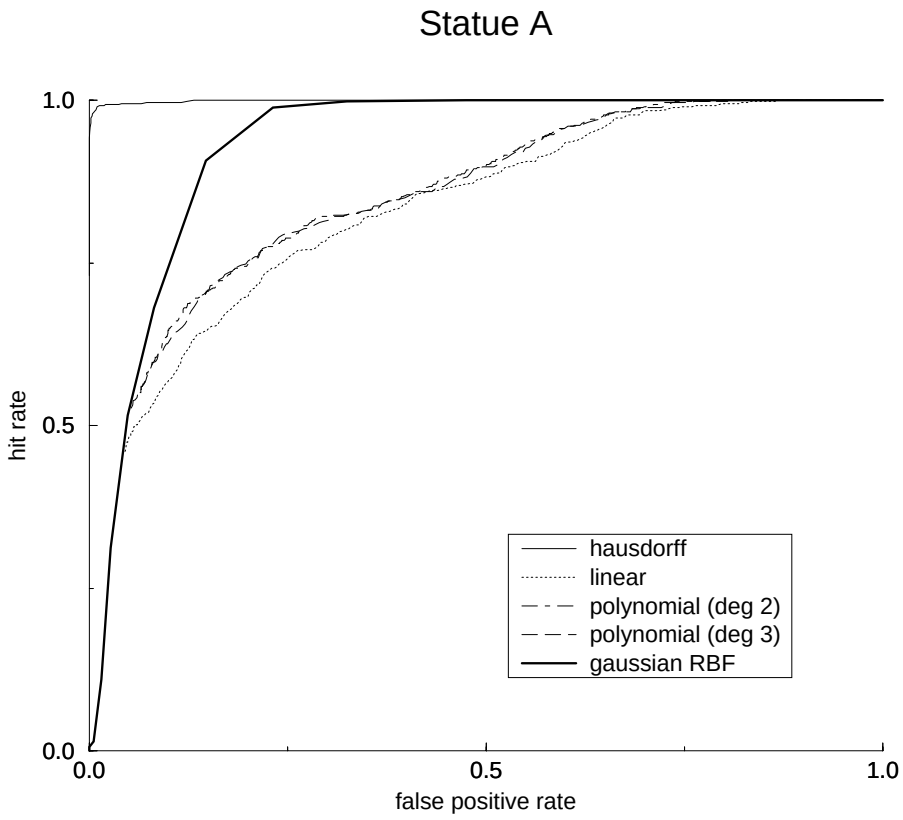
All the images were acquired in the same location and thus have a similar background. No background elimination was performed since the face occupies a substantial part of each image — about three quarters of the total image area, — but this implies that even images of different people have, on average, one quarter of the pixels which match. This is shown in Figure 7.10, where the white pixels in the rightmost images are points of the first image close in the Hausdorff sense to the middle image — they correspond mainly to background pixels. Notice that the two binary images are similar even though one of them has been computed by comparing two images of the same subject, but the other one by comparing two images of different subjects. It is thus clear that, in this context, one or a few images are not sufficient for object characterization.

Figure 7.11 shows the ROC curves of the Hausdorff kernel for all the four face recognition systems, and a comparison with classical kernels (linear, polynomial of degree 2, and Gaussian). Including the linear kernel in this comparison might sound simplistic, but this choice it due to the fact that the linear kernel is similar to classical correlation techniques (such as sum of squared differences and cross correlation), as shown in Section 7.2. The curve obtained with the Hausdorff kernel is always above the others, showing superior performance. The ROC curve corresponding to the Hausdorff kernel increases rapidly and shows good properties of sensitivity and specificity. By contrast, linear and polynomial kernels do not appear to be suitable for the task: in all the four cases, to obtain a hit rate of 90% with the linear kernel one would have to accept more than 50% false positives. Gaussian kernel shows better performances, just below the Hausdorff kernel. We tested the robustness of the recognition system by adding difficult positives to one of the test sets (see Figure 7.12). The corresponding ROC curve is the one in the lower left corner of Figure 7.11.

In a second series of experiments on the same training sets, we estimated the system performance in the *leave-one-out* mode — that is, we used all possible $\ell - 1$ data points for training and tested the system generalization capability with the data left out. Figure 7.13 shows samples from one of the training sets. Even though the set contains several images that are difficult to classify, only 19% were found to lie outside the sphere of minimum radius, and none of them by a distance of more than 3% of the estimated radius.

Second data set

Other sets of face images were acquired on different days and under unconstrained illumination (see examples in Figure 7.14). A significant change of scale is immediately noticeable comparing Figure 7.8 and Figure 7.14, which therefore necessitated background elimination from the training data. To this purpose, we performed a semi-automatic preprocessing of the training data, exploiting the spatio-temporal continuity between adjacent images of the training set: in fact, we manually selected a rectangular patch in the first image of each sequence, and then tracked it automatically through the rest of the sequence, thus obtaining the reduced images shown in Figure 7.15 which we used to train our system to recognize this specific subject (Andrea). Figures 7.16 and 7.17 respectively show positive and negative test images, used to test the system. The performance of the system in recognizing Andrea among examples of different people is described by the ROC curve at the top right of Figure 7.18. Figure 7.18 also shows the performance of the other three systems. The four different training sets were composed of 126, 89, 45, 40 images (corresponding to the curves in clockwise order, from the top left). The size of the test set is specified in the caption of the figure. With this second class of data sets, the results of the linear kernel were in most cases too poor to represent a good comparison, so we also experimented with polynomial kernels of various degrees. In the ROCs of Figure 7.18 we include the results obtained with a polynomial kernel of degree 2, the one which produced the best results. One of the reasons for the failure of classic polynomial kernels may be related to the fact that the data we use are not correlated, i.e., the images have not been accurately registered with respect to a common reference. During the acquisition the person was moving and, in different images, the features (eyes, for instance, in the case of faces) are set in different positions.



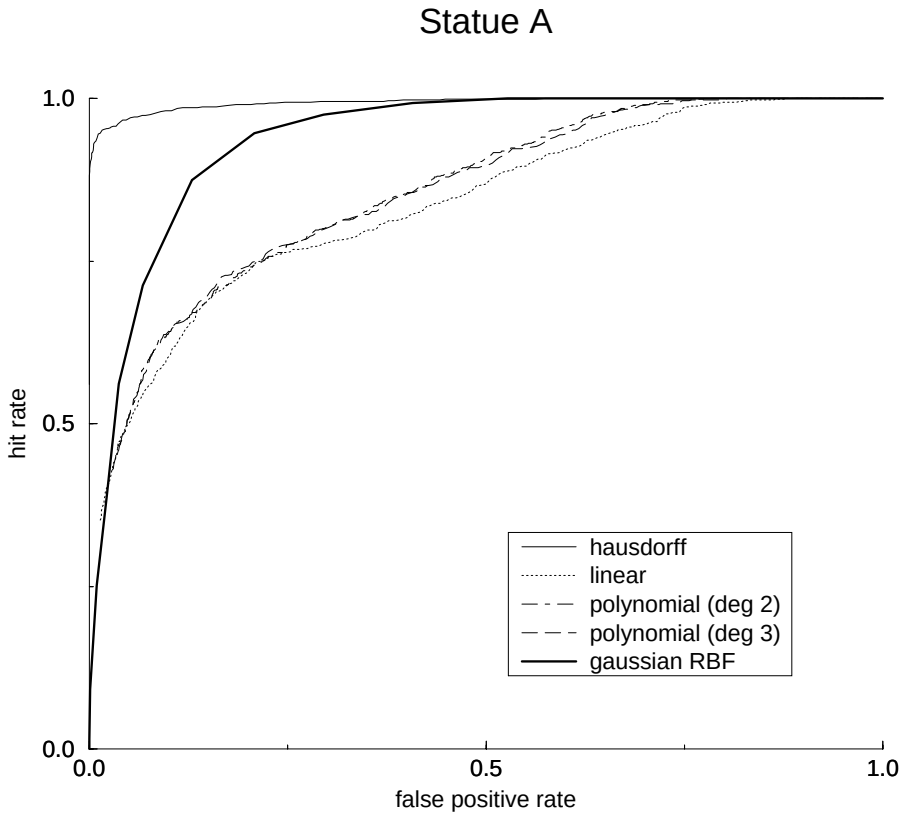
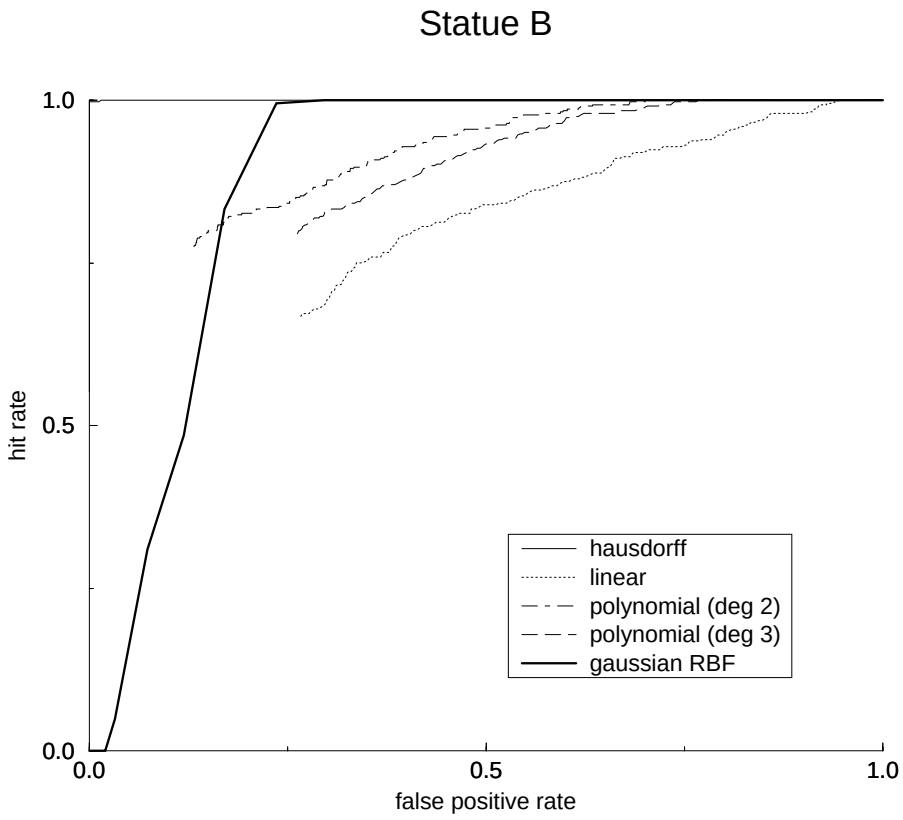


Figure 7.5: The effect of decreasing the size of the training set, on the system for statue A, from 314 (above) down to 96 elements (below). Size of the test set of positive examples: on the above 628, on the below 846. Size of the test set for negative examples: 3500 for both experiments.



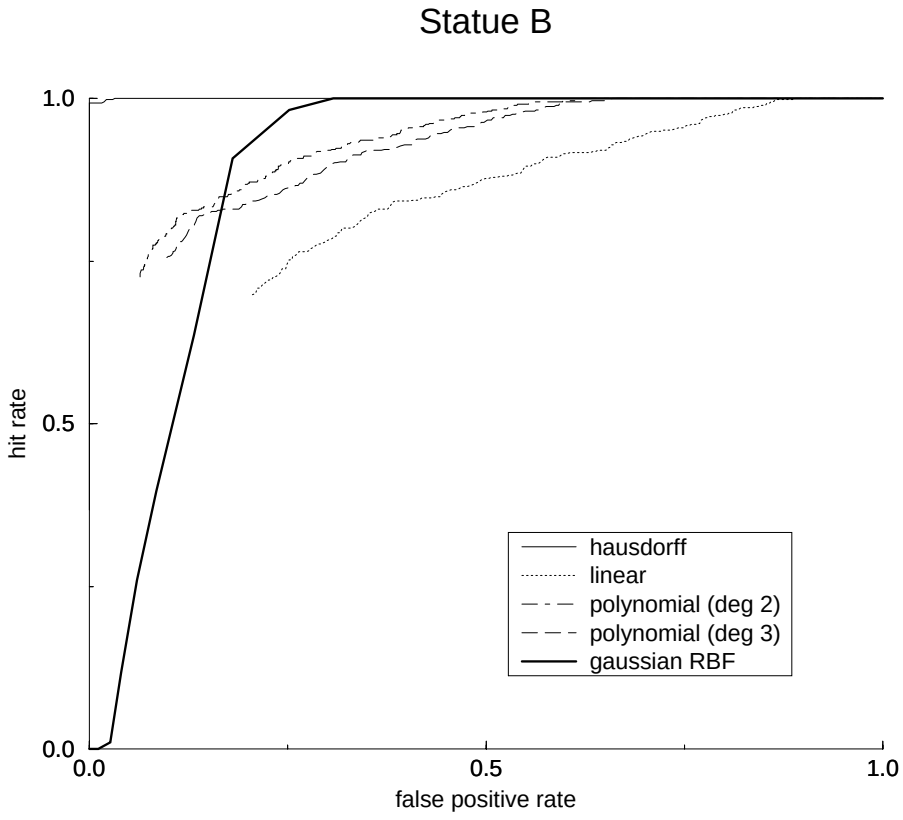


Figure 7.6: The effect of decreasing the size of the training set, on the system for statueB, from 226 (above) down to 97 elements (below). Size of the test set of positive examples: on the above 449, on the below 577. Size of the test set for negative examples: 3500 for both experiments.

Figure 7.7: Identification of statue A and statue B within a sequence: the white rectangles are the matches of statue A, the grey rectangles the matches of statue B.

Figure 7.8: Two training images for each of the four subjects.

Figure 7.9: Examples of test images.

Figure 7.10: Spatial support of the Hausdorff distance. Both rows: the white pixels in the rightmost image show the locations of the leftmost image which are close, in the Hausdorff sense, to the middle image.

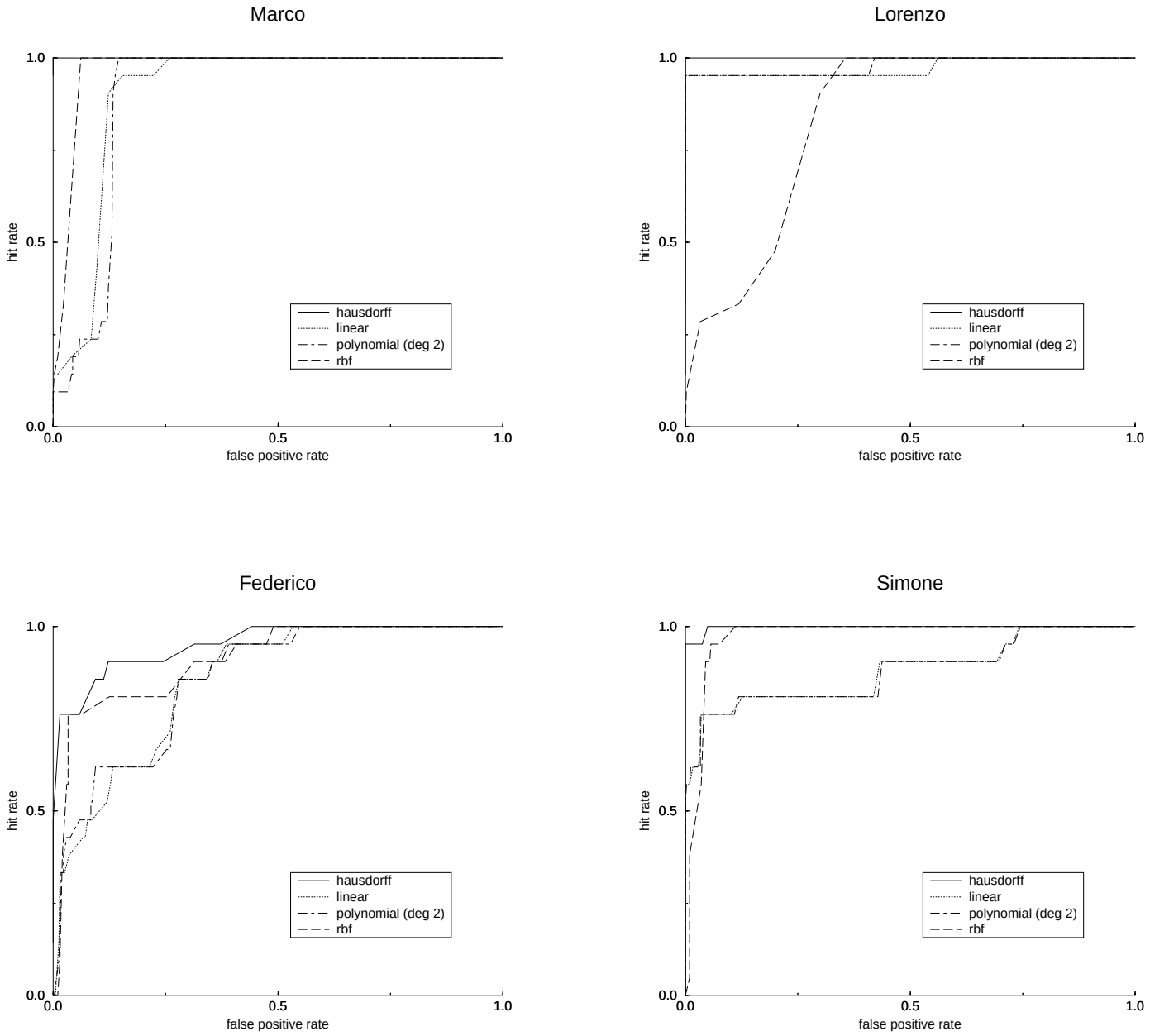


Figure 7.11: ROC curves for the four training sets. Comparison between linear kernel, polynomial kernel (degree 2), Gaussian kernel and Hausdorff kernel. From top-left, in clockwise order: Marco, Lorenzo, Simone, Federico.

Figure 7.12: Examples of the images which affected the ROC curve in the lower left corner of Figure 7.11.

Figure 7.13: Examples of the data set used for the leave one out experiment.

Figure 7.14: One image for each one of the subjects of the new experiments.

Figure 7.15: Face tracking throughout the sequence. Top rows: original sequences (acquired on two different days); bottom rows: reduced images.

Figure 7.16: Samples from positive test sequences, relative to the training set of Figure 7.15.

Figure 7.17: Samples from negative test sequences, relative to the the training set of Figure 7.15.

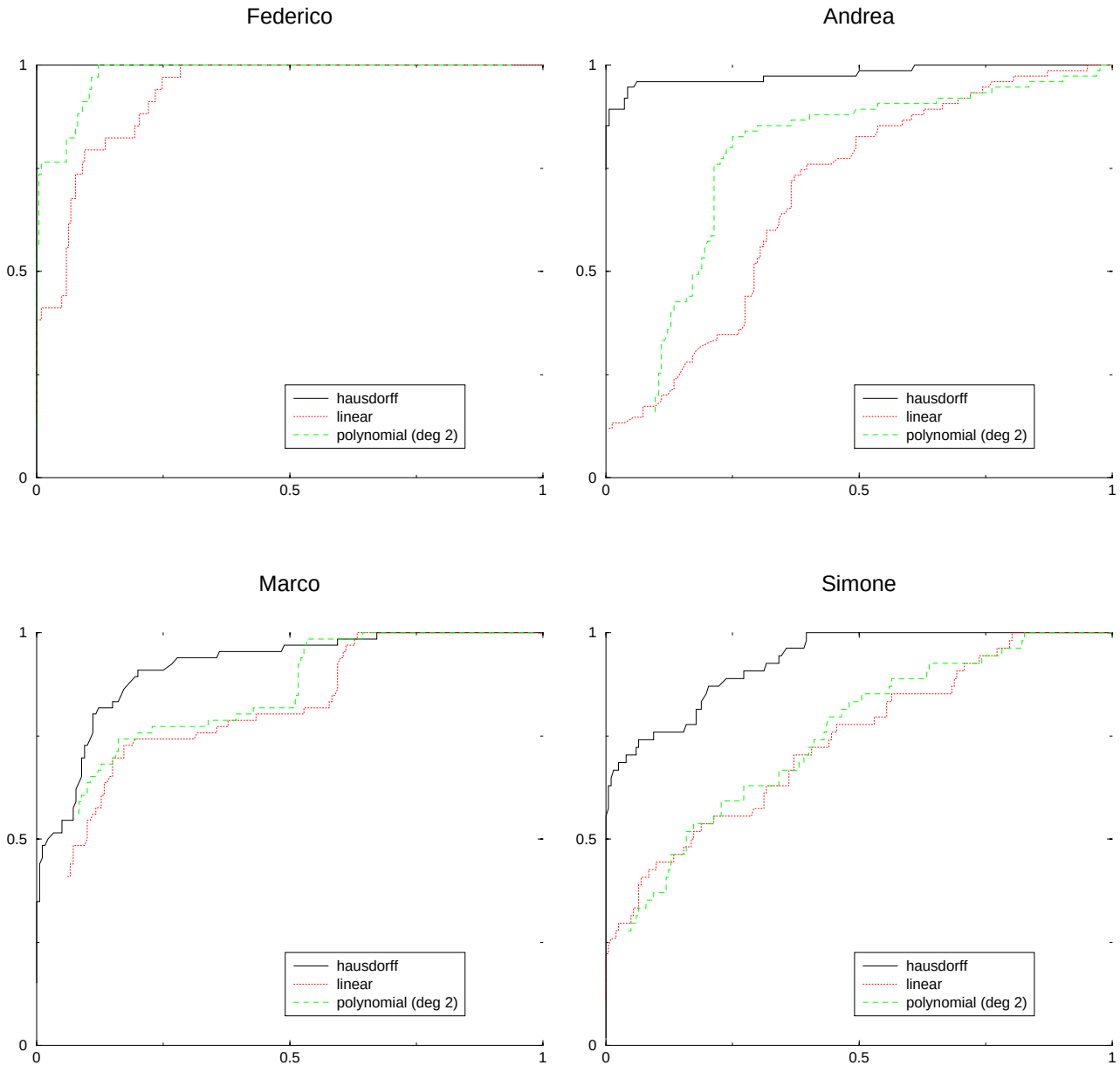


Figure 7.18: ROC curves the four subjects of the second data set, comparing the Hausdorff kernel, with linear and polynomial (deg. 2) ones. From top left: Federico (training: 126, test: 102 (positive), 205 (negative)); Andrea (training; 89, test: 228 (positive) 345 (negative)); Marco (training: 40, test: 149 (positive), 343 (negative)); Simone(training; 45, test: 122 (positive), 212 (negative)).

Discussion of Part III

In Part III we addressed the problem of representing and identifying 3D objects from a set of positive examples only: thus we approached a multiclass classification problem by learning one class at a time — when used on a collection of significant views of the same object, this approach could be regarded as a method for object modeling.

We used a kernel-based method which bears some resemblance to Support Vector Machines for binary classification, and we equipped it with a function specifically designed to deal with grey level images, that in the one class learning case can be used as a kernel. This function, which we derived from the Hausdorff-based similarity measure introduced in Chapter 5, tolerates small illumination and scale variations, and, by construction, is tolerant to small spatial misalignments. For this reason we could achieve good representation results while keeping the preprocessing of input data minimal, and, unlike other methods for representing 3D object with 2D images as morphable models, we did not require any preliminar registration of input data.

The method proposed was used to model faces of different subjects and to perform face identification experiments. Also, we used it to represent a difficult collection of marble statues (which are rather similar one to the other) and to perform 3D object recognition in single novel views.

Our current work on this subject ranges from theoretical to applied aspects of this object representation, and in the Section Conclusions and Future Work we will describe the current state of the work, and the plans for future developments.

Conclusions and Future Work

This thesis has addressed the problem of representing 3D objects from image sequences, for object identification and recognition tasks. The ambition of our approach is to be able to capture an unconstrained image sequence of an object — simply by moving a camera around it — and to use it for representing the object, while keeping the amount of analysis and of preprocessing low. This is a broad, difficult problem: this thesis was mainly devoted to studying it, and to delineating and addressing some of its main aspects that fall in the fields of computer vision and machine learning.

The results obtained represent contributions in the following aspects of object representation and identification from image sequences. Firstly, we presented a robust sequence registration technique (Chapter 2) and applied it to both mosaic construction and mosaic-based multiple motion segmentation (Chapter 3). We used this technique to stabilize video sequences which were intended to be still, and to perform video compression, thus enhancing the quality of some sequence on the one hand, and reducing the redundancy of sequences on the other. Secondly, we introduced a multiscale similarity method based on the definition of Hausdorff distance (Chapter 5), which is tolerant to scene variations (that may induce occlusions), to small light variations, and performs well even in the presence of small geometric deformations. We used this method as a *brute-force* tool for object identification, which is especially effective when used in controlled environments (Chapter 5), as an object tracking method which is useful for a semi-automatic size reduction of image sequences (Chapter 7), and, within the framework of kernel-based learning methods, as a kernel function for images (Chapter 7). Thirdly, we studied the problem of representing 3D objects by means of a collection of 2D unconstrained images. We approached

this problem by means of a kernel-based classification system that learns one class at a time (i.e., that models one object at a time) using positive examples only. Finally, we analyzed the properties of the Hausdorff-based similarity measure introduced in Chapter 5 as a kernel function: we noted that, thanks to its intrinsic tolerance of spatial misalignments, we could reduce the preprocessing of input data, avoiding the registration of all the images of the training set.

Each contribution introduced in this thesis is somewhat independent from the others and it can be used to solve specific applications but, taken all together, they can represent the main tools for a complex object representation and identification system. The latter constitutes the main thrust of our current activity. From the point of view of the application, we are developing a prototype for a face identification system which will be installed at the entrance of our laboratory. This system assumes a collaborative and constrained environment, where the people will aim at being recognized and the background and illumination will be controlled. At present we are addressing the problem with a brute-force approach: each subject is represented by a small (less than 10) collection of image models, and the recognition task is performed on an image sequence, by applying on each frame (and with respect to each model) the multiscale search method introduced in Chapter 5. The main problems here are time and abrupt variations of test images from all the representative models (which may be caused by a change in the particular person, of the hair style or the glasses for instance). In the future, we will also combine into to the system the statistical learning method for learning one class at a time. This should allow us to describe each subject by a more complete selection of image models.

We are also studying the problem of using our 3D object recognition system as a support for navigation in complex environments, such as locations of artistic or architectural relevance. Our methodology for object representation will be used to describe objects of interest, while our method for object identification from a single novel view will be extended to deal with image sequences. This problem, which can be seen as a video querying problem, present several open issues. The one we are currently addressing is the analysis of a test sequence in order to identify representative sub-sequences or shots to use for querying (thus avoiding redundancy of the queries). There are various possi-

ble heuristics that can be applied to address this problem: we are studying the typology of the camera motion to discriminate interesting shots from the rest of the video. To investigate this problem, we have made some simplistic choices about the types of camera motion that would make a portion of the sequence “interesting”, but the results obtained so far, even if very preliminary, are promising. In practical terms, this technique can be used to preprocess the video stream for video querying in the following way. If the user is interested in a specific object we assume that he will stand by it for longer, he will get close to the object, or he will perhaps keep the camera still for a few seconds. Even if this reasoning is too simplistic, it is true that a rough motion estimate can give us useful information about which part of the video most deserves the focus of attention.

From a theoretical point of view, the kernel-based novelty detection method of Chapter 7 presents various open issues. Firstly, the problem could be reformulated as a regularization problem, similarly to what has been done for Support Vector Machines[EPP00b, Wah90]: we believe that to address this problem will require a deeper understanding of the relationship between the decision surfaces of binary classification and the sphere of one class classification.

Credits

The year I spent at Heriot-Watt University (Edinburgh), where I developed Part I of this thesis, was supported by a Marie Curie Training and Mobility Reasearch grant.

The software of the thesis was coded in ANSI C, and compiled using the public domain `gcc` compiler. Some programs include numerical routines from Numerical Recipes [PTVF88].
It was written by myself, occasionally in collaboration with other people; in particular:

- The code for mosaicing and image registration was written together with Andrea Fusiello, and the corner tracking code we used was written by Costas Plakas;
- The multiresolution version of the Hausdorff similarity method was written by Simone Ursino;
- The SVM adaption to novelty detection is mainly due to Annalisa Barla and myself;
- As concerns the SVM code, it exists thanks to the contributions of a number of people, but I think we all agree that the "father" is Massimiliano Pontil.

As concerns the images included in this thesis, they have been acquired *in loco* [1], inside the Dipartimento di Informatica e Scienze dell'Informazione, in the center of Genova (Piazza Matteotti, Piazza San Lorenzo) or inside San Lorenzo Cathedral (with the permission of the Curia), with the exception of:

[1] They are all available for download. See `http://www.disi.unige.it/person/OdoneF`.

- The stereo pairs used in Chapter 5, which are part of the JISCT (JPL-INRIA-SRI-CMU-TELEOS) stereo test set [2].
- The sequence "Affresco" used in Chapter 3, which is courtesy of Mr Anthony Doull.
- The image sequences [3] used for the experiments of Section 3.4, which were acquired at the Heriot-Watt University (Edinburgh).

The "models" used for the face identification experiments are: Elena Balladore, Annalisa Barla, Luisa Berutti, Giada Carlini, Francesco Isgrò and his sole expression, Federico Fassone, Andrea Del Fiandra, Marco Razeto, Lorenzo Rosasco and all his expressions (see Figure 7.13), and Simone Ursino.

The work presented in this thesis has benefited from discussions with various people: Alessandro Verri, Annalisa Barla, Andrea Fusiello, Tim Heap, Francesco Isgrò, Yvan Petillot, Emanuele Trucco.

L'inglese come lo scrivono a Cambridge è dovuto a Tim Heap.

[2] Available at `ftp://ftp.vislist.com/IMAGERY/JISCT`.
[3] Available for download at `http://www.disi.unige.it/person/OdoneF/mosaic`.

Bibliography

[Ade91] E. H. Adelson. Layered representations for image coding. Technical Report 181, MIT Media Lab Vision and Modeling Group, 1991.

[BBHP92] J.R. Bergen, P.J. Burt, R. Hingorani, and S. Peleg. A three frame algorithm for estimating two component image motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1992.

[BBW88] P. Besl, J. Birch, and L. Watson. Robust window operators. In *2nd International Conference on Computer Vision*, pages 591–600, 1988.

[BCB98] G. Baldi, C. Colombo, and A. Del Bimbo. Automatic video representation using mosaicing. In R. Cucchiara and M. Piccardi, editors, *Proceedings of the Joint Workshop of AI*IA and IAPR-IC*, pages 152–157, Ferrara, April 1998. Università di Ferrara - Facoltà di Ingegneria.

[BET95] H. H. Bulthoff, S. Y. Edelman, and M. J. Tarr. How are three-dimensional objects represented in the brain? *Cerebral Cortex*, 5(3), 1995.

[BFB94] J. L. Barron, D. J. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[BJ85] P. J. Besl and R. C. Jain. Three-dimensional object recognition. *Computing Surveys*, 17(1):75–145, 1985.

[BMM99] R. Brunelli, O. Mich, and C. M. Modena. A survey on the automatic indexing of video data. *Journal of Visual Communication and Image Representation*, 10:78–112, 1999.

[BN98] D. N. Bhat and S. K. Nayar. Ordinal measures for image correspondence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20, 1998.

[Bor84] G. Borgefors. Distance transforms in arbitrary dimensions. *Computer Vision, Graphics and Image Processing*, 27:321–345, 1984.

[BOV02] A. Barla, F. Odone, and A. Verri. Hausdorff kernel for 3d object acquisition and detection. In *Proceedings of the European Conference on Computer Vision*, Copenhagen, Denmark, May 2002.

[Bro92] L. G. Brown. A survey on image registration techniques. *ACM Computing Surveys*, 24(4):325–376, December 1992.

[BS79] M. Bazaraa and C. M. Shetty. *Nonlinear programming*. John Wiley, New York, 1979.

[BS00] H. Burkhardt and S. Siggelkow. Invariant features in pattern recognition - fundamentals and applications. In C. Kotropoulos and I. Pitas, editors, *Nonlinear model-based image/video processing and analysis*. John Wiley & sons, 2000.

[BSF88] Y. Bar-Shalom and T. E. Fortmann. *Tracking and data Association*. Academic Press, 1988.

[BSP93] D. Beymer, A. Shashua, and T. Poggio. Example based image analysis and synthesis. Technical Report A.I. Memo No. 1431, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1993.

[Bur98] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[BVZ98] Y. Boykov, O. Veksler, and R. Zabih. A variable window approach to early vision. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 20(12), December 1998.

[CB01] C. Campbell and K. P. Bennett. A linear programming approach to novelty detection. *Advances in Neural Information Processing Systems*, 13, 2001.

[CCM+97] S. F. Chang, W. Chen, H. J. Meng, H. Sundaram, and D. Zhong. Videoq: An automated content based video search using visual cues. In *Fifth ACM Multimedia Conference*, Seattle, November 1997.

[CH62] R. Courant and D. Hilbert. *Methods of mathematical physics, Vol 2*. Interscience, London, UK, 1962.

[CHV99] I. Chapelle, P. Haffner, and V. Vapnik. SVMs for histogram-based image classification. *IEEE Transactions on Neural Networks, special issue on Support Vectors*, 1999.

[CM99] I. Cohen and G. Medioni. Detecting and tracking moving objects in video surveillance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II:319–325, 1999.

[CST00] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

[CV92] M. Campani and A. Verri. Motion analysis from first order properties of optical flow. *Computer Vision, Graphics, and Image Processing*, 56(1):90–107, 1992.

[CV95] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[CWT00] T. F. Cootes, K. N. Walker, and C. J. Taylor. View-based active appearance models. In *IV International Conference on Automatic Face and Gesture Recognition*, pages 227–232, 2000.

[CZ98] David Capel and Andrew Zisserman. Automated mosaicing with super-resolution zoom. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 885–891, 1998.

[DB93] R. Deriche and T. Blaszka. Recovering and characterizing image features using an efficient model based approach. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 530–535, 1993.

[DC95] P. Dani and S. Chaudhuri. Automatic assembling of images: image montage preparation. *Pattern Recognition*, 28(3):431–445, 1995.

[DR96] Z. Durich and A. Rosenfeld. Image sequence stabilization in real-time. *Real-TIme Imaging*, 2:271–284, 1996.

[DY99] D.Chetverikov and Y.Khenokh. Matching for shape defect detection. In *Lecture Notes in Computer Science*, volume 1689, pages 367–374, 1999.

[ECT99] G. Edwards, T. F. Cootes, and C. J. Taylor. Advances in active appearance models. In *VII International Conference on Computer Vision*, 1999.

[EPP00a] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and Support Vector Machines. *Advances in Computational Mathematics*, 13:1–50, 2000.

[EPP00b] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–50, 2000.

[FRT97] A. Fusiello, V. Roberto, and E. Trucco. Efficient stereo with multiple windowing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 858–863, Puerto Rico, June 1997. IEEE Computer Society Press.

[FTTR99] A. Fusiello, E. Trucco, T. Tommasini, and V. Roberto. Improving feature tracking with robust statistics. *Pattern Analysis and Applications*, 2(4):312–320, 1999.

[GJ98] P.R. Giaccone and G.A. Jones. Segmentation of global motion using temporal probabilistic classification. In *British Machine Vision Conference*, pages 619–628, 1998.

[GJP95] F. Girosi, M. Jones, and T. Poggio. Regularization theory and neural networks architectures. *Neural computation*, 7:219–269, 1995.

[GL96] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The John Hopkins University Press, third edition, 1996.

[GLK00] G. Guodong, S. Li, and C. Kapluk. Face recognition by support vector machines. In *Proc. IEEE International Conference on Automatic Face and Gesture Recognition*, pages 196–201, 2000.

[GM99] A. A. Goshtasby and J. Le Moigne, editors. *Pattern Recognition. Special Issue: Image Registration*, volume 32. Pergamon, January 1999.

[GS74] D. M. Green and J. A. Swets. *Signal Detection Theory and Psycophysics*. Wiley, Chicester New York Brisbane Toronto, 1966, Reprinted by Krieger, Huntingdon, New York,, 1974.

[HAD+94] M. Hansen, P. Anandan, K. Data, G. Wal, and P. Burt. Real-time scene stabilization and mosaic construction. In *Proceedings of IEEE Workshop on Applications of Computer Vision*, 1994.

[Har95] R. I. Hartley. In defence of the 8-point algorithm. In *Proceedings of the IEEE International Conference on Computer Vision*, 1995.

[HE98] P. Hastreiter and T. Ertl. Integrated registration and visualization of medical image data. In *Proceedings of COmputer Graphics International (CGI)*, 1998.

[HKR93] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. Comparing images using the hausdorff distance. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 9(15):850–863, 1993.

[HKS91] D. P. Huttenlocher, K. Kedem, and M. Sharir. The upper envelope of voronoi surfaces and its applications. In *Proceedings of Seventh ACM Symposium of Computational Geometry*, pages 194–293, 1991.

[HR93] D. P. Huttenlocher and W. J. Rucklidge. Multi-resolution technique for comparing images using the hausdorff distance. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 705–706, 1993.

[HS81] B. K. P. Horn and B. G. Schunk. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[HSPP01] B. Heisele, T. Serre, M. Pontil, and T. Poggio. Component-based face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 657–662, 2001.

[HZ00] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.

[IA96] M. Irani and P. Anandan. Parallax geometry of pairs of points for 3D scene analysis. In *Proceedings of the European Conference on Computer Vision*, pages 17–30, 1996.

[IAB+96] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S Hsu. Efficient representations of video sequences and their applications. *Signal processing: Image Communication*, 8(4), 1996.

[IAH95] M. Irani, P. Anandan, and S. Hsu. Mosaic based representations of video sequences and their applications. In *International Conference on Computer Vision*, pages 605–611, 1995.

[IHA98] M. Irani, S. Hsu, and P. Anandan. Video indexing based on mosaic representations. In *Proceedings of the IEEE*, volume 86:5, pages 905–921, 1998.

[IRP94a] M. Irani, B. Rousso, and S. Peleg. Computing occluding and transparent motions. *International Journal of Computer Vision*, 12(1):5–16, February 1994.

[IRP94b] Michael Irani, Benny Rousso, and Shmuel Peleg. Recovery of ego-motion using image-stabilization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[Isg01] F. Isgrò. *Geometric Methods for Video Sequence Analysis and Applications*. PhD thesis, Heriot-Watt University, Department of Computing and Electrical Engineering, 2001.

[JMKL00] K. Jonsson, J. Matas, J. Kittler, and Y. Li. Learning support vectors for face verification and recognition. In *Proc. IEEE International Conference on Automatic Face and Gesture Recognition*, 2000.

[Joa98] T. Joachims. Text categorization with Support Vector Machine. In *Proceedings of the European Conference on Machine Learning*, 1998.

[JP98] M. J. Jones and T. Poggio. Multidimensional morphable models: a framework for representing and matching object classes. *International Journal of Computer Vision*, 2(29):107–131, 1998.

[KO94] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: theory and esperiment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):920–932, September 1994.

[KPC97] R. Koenen, F. Pereira, and L. Chiariglione. MPEG-4: Context and objectives. *Signal Processing: Image Communications*, 9(4):295–, 1997.

[KS99] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. In *Proceedings of the 7th International Conference of Computer Vision*, pages 307–314, 1999.

[KSPF96] R. Kahn, M. Swain, P. Prokopowicz, and R. Firby. Gesture recognition using the perseus architecture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 734–741, 1996.

[Kut00] K. N. Kutulakos. Approximate N-view stereo. In D. Vernon, editor, *Proceedings of the 6th European Conference on Computer Vision*, volume 1 of *LNCS 1842*, pages 67–81. Springer, 2000.

[KvD79] J. J. Koenderink and A. J. van Doorn. The internal representation of solid shape with respect to vision. *Biol. Cyber.*, 24:211–216, 1979.

[LH81] H. C. Longuett-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(10):133–135, September 1981.

[LHP80] H. C. Longhett-Higgins and K. Prazdny. The interpretation of a moving retinal image. *Proceedings of the Royal Society of London B*, 208:385–397, 1980.

[Li92] S. Li. Matching: invariant to translations, rotations and scale changes. *Pattern Recognition*, 25(6):583–594, 1992.

[LZ98] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 482–488, 1998.

[MB96] M. Massey and W. Bender. Salient stills: process and practice. *IBM Systems Journal*, 35(3,4), 1996.

[MC96] C. H. Morimoto and R. Chellappa. Fast electronic digital image stabilization. In *Proceedings of the International Conference on Pattern Recognition*, Austria, Vienna, 1996.

[MMRK91] P. Meer, D. Mintz, A. Rosenfeld, and D. Kim. Robust regression methods for computer vision: a review. *International Journal of Computer Vision*, 6(1):59–70, 1991.

[MP94] S. Mann and R. W. Picard. Virtual bellows: Contructing high quality stills from video. In *IEEE International Conference on Image Processing*, 1994.

[MPP01] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *IEEE Trans. Patt. Anal. Mach. Intell.*, 23:349–361, 2001.

[NNT01] E. Sánchez Nielsen, J. Lorenzo Navarro, and M. Hernández Tejera. Increasing efficiency of hausdorff approach for tracking real scenes with complex environments. In *Proceedings of the International Conference on Image Analysis and Processing*, pages 131–136, Palermo, Italy, September 2001.

[OFT00] F. Odone, A. Fusiello, and E. Trucco. Robust motion segmentation for content-based video coding. In *6th RIAO (Recherche d'Informations Assiste par Ordinateur) International Conference*, 2000.

[OFT01] F. Odone, A. Fusiello, and E. Trucco. A layered representation of a video shot with mosaicing. *Pattern Analysis and Applications*, 2001. To appear.

[Ols97] C. F. Olson. Mobile robot self-localization by iconic matching of range maps. In *Proceedings of the Int. Conf. on Advanced Robotics*, pages 447–452q, 1997.

[Ols98] C. F. Olson. A probabilistic formulation for hausdorff matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 150–156, 1998.

[OTV01a] F. Odone, E. Trucco, and A. Verri. A flexible algorithm for image matching. In *Proceedings of the International Conference on Image Analysis and Processing"*, pages 290–295, Palermo, Italy, September 2001.

[OTV01b] F. Odone, E. Trucco, and A. Verri. General purpose matching of grey level arbitrary images. In G. Sanniti di Baja C. Arcelli, L. P. Cordella, editor, *4th International Workshop on Visual Forms*, Lecture Notes on Computer Science LNCS 2059, pages 573–582. Springer, 2001.

[Pol] M. Pollefeys. 3-D modeling from images. in conjunction with ECCV 2000.

[PP00] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.

[PS85] F. P. Preparata and M. I. Shamos. *Computational Geometry. An Introduction*, chapter 2, pages 72–77. Springer-Verlag, first edition, 1985.

[PTVF88] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C.* Cambridge University Press, second edition, 1988.

[PV92] T. Poggio and T. Vetter. Recognition and structure from one 2D model view: observations on prototypes, object classes and symmetries. Technical Report A.I. Memo 1347, MIT, 1992.

[PV98] M. Pontil and A. Verri. Mathematical properties of support vector. *Neural Computation*, 10:977–996, 1998.

[PZ98] P. Pritchett and A. Zisserman. Wide baseline stereo matching. In *Proc. 6th International Conference on Computer Vision, Bombay, India*, pages 754–760, January 1998.

[RBK98] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:23–38, 1998.

[RJV99] T. D. Rikbert, M. J. Jones, and P. Viola. A cluster-based statistical model for object detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Fort Collins, 1999.

[RL87] P. J. Rousseeuw and A. M. Leroy. *Robust regression & outlier detection*. John Wiley & sons, 1987.

[RPFRA98] B. Rousso, S. Peleg, I. Finci, and A. Rav-Acha. Universal mosaicing using pipe projection. In *International Conference on Computer Visione (ICCV98)*, 1998.

[Ruc97] W. J. Rucklidge. Efficiently locating objects using the hausdorff distance. *Int. Journ. of Computer Vision*, 24(3):251–270, 1997.

[SA96] H. Sawhney and S. Ayer. Compact representations of videos through dominant and multiple motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):814–830, August 1996.

[SBV95] B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In U. M. Fayyad and R. Uthurusamy, editors, *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 252–257. AAAI Press, 1995.

[SD99] R. Sim and G. Dudek. Learning and evaluating visual features for pose estimation. In *ICCV99*, pages 1217–1222, 1999.

[Ser82] J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, 1982.

[Sin95] P. Sinha. *Perceiving and recognizing 3D forms*. PhD thesis, Massachussetts Institute of Technology, 1995.

[SK52] J. G. Semple and G. T. Kneebone. *Algebraic projective geometry*. Oxford University Press, 1952.

[SK00] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[SM97] C. Schmid and R. Mohr. Local greyvalue invariants for image retrieval. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 19(5):530–534, May 1997.

[SN96] A. Shashua and N. Navab. Relative affine structure: Canonical model for 3D from 2D geometry and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):873–883, September 1996.

[SP97] R. Sharma and M. Pavel. Registration of video sequences from multiple sensors. In *R. K. Sharma and M. Pavel, Registration of video sequences from multiple sensors, in Proceedings of the Image Registration Workshop*, pages 361–366, 1997.

[SPST+00] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. Technical Report MSR-TR-99-87, Microsoft Research Corporation, 1999,2000.

[ST93] J. Shi and C. Tomasi. Good features to track. Technical Report 93-1399, Department of Computer Science, Cornell University, Ithaca, NY 14853-7501, November 1993.

[ST94] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, June 1994.

[Sze96] R. Szeliski. Video mosaics for virtual environments. *IEEE Computer Graphics and Applications*, 16(2):22–30, March 1996.

[TD99] D. Tax and R. Duin. Data domain description by support vectors. In M. Verleysen, editor, *Proceedings of ESANN99*, pages 251–256. D. Facto Press, 1999.

[TK91] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, Pittsburg, PA, April 1991.

[TPR+00] E. Trucco, Y. R. Petillot, I. Tena Ruiz, K. Plakas, and D.M. Lane. Feature tracking in video and sonar subsea sequences with applications. *Computer Vision and Image Understanding, special issue on Underwater Computer Vision and Pattern Recognition*, 2000. Accepted for publication.

[TV98] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice-Hall, 1998.

[TYD99] D. Tax, A. Ypma, and R. Duin. Support vector data description applied to machine vibration analysis. In *Proc. of the 5th Annual Conference of the Advanced School for Computing and Imaging*, 1999.

[Vap95] V. Vapnik. *The nature of statistical learning Theory*. John Wiley and sons, New York, 1995.

[Vap98] V. Vapnik. *Statistical learning theory*. John Wiley and sons, New York, 1998.

[VP97] T. Vetter and T. Poggio. Linear object classes and image synthesis from a single example image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):733–742, 1997.

[WA94] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3(5):625–638, May 1994.

[Wah90] G. Wahba. *Spline models for observational data*. SIAM, 1990.

[WFKvdM97] L. Wiskott, J. M. Fellous, N. Krüger, and C. von der Malsburg. A statistical method for 3D object detection applied to faces and cars. In *Proc. IEEE Int. Conf. on Image Processing*, Wien, 1997.

[ZFD97] I. Zoghlami, O. Faugeras, and R. Deriche. Using geometric corners to build a 2D mosaic from a set of images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 420–425, 1997.

[Zha97] Z. Zhang. Parameter estimation techniques: a tutorial with application to conic fitting. *Image and Vision Computing*, 15(1):59–76, 1997.

[ZW94] R. Zabih and J. Woodfill. Non-parametric local transformas for computing visual correspondence. In *Proceedings of the third European Conference on Computer Vision*, volume 2, 1994.